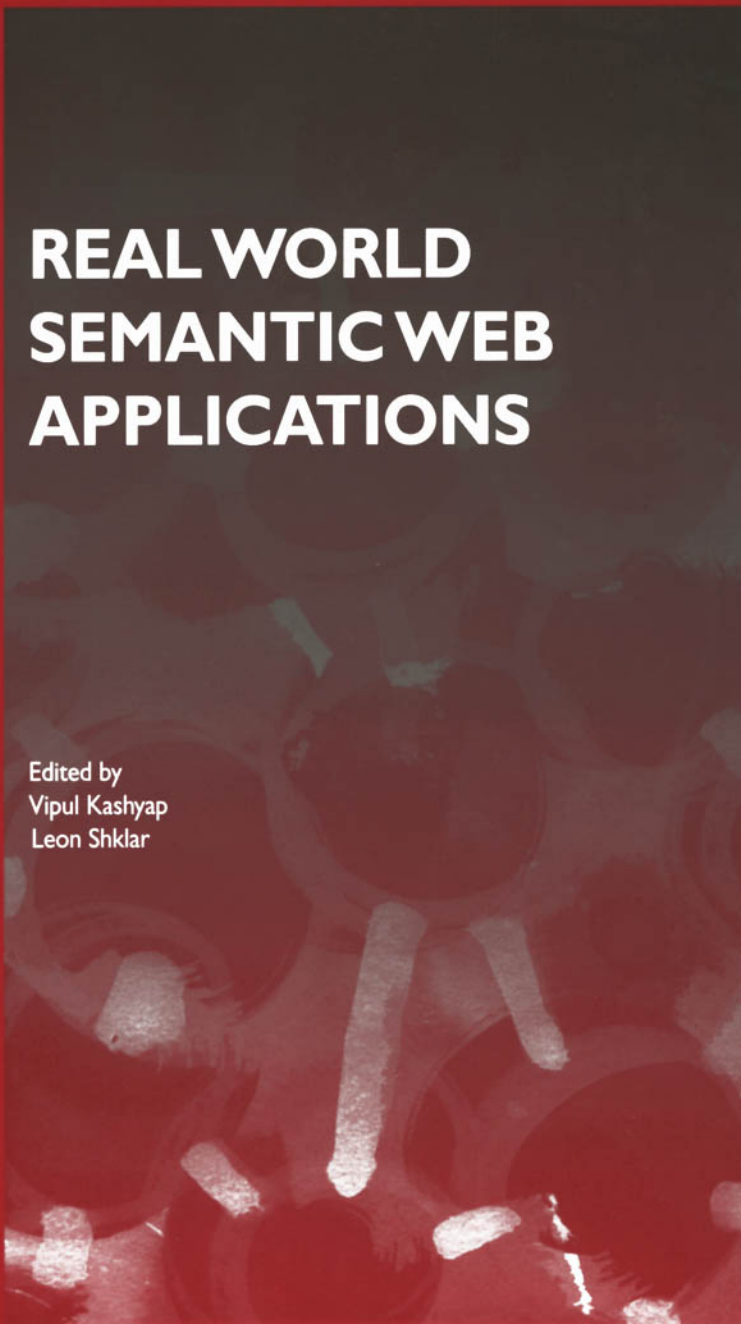
Frontiers in Artificial Intelligence and Applications

REAL WORLD SEMANTIC WEB APPLICATIONS

Edited by
Vipul Kashyap
Leon Shklar



IOS Press
Ohmsha

REAL WORLD SEMANTIC WEB APPLICATIONS

Frontiers in Artificial Intelligence and Applications

Series Editors: J. Breuker, R. López de Mántaras, M. Mohammadian, S. Ohsuga and W. Swartout

Volume 92

Recently published in this series:

Vol. 91, F. Azevedo, Constraint Solving over Multi-valued Logics – Application to Digital Circuits
Vol. 90, In production
Vol. 89, In production
Vol. 88, In production
Vol. 87, A. Abraham et al. (Eds.), Soft Computing Systems: Design, Management and Applications
Vol. 86, In production
Vol. 85, J.M. Abe and J.I. da Silva Filho (Eds.), Advances in Logic, Artificial Intelligence and Robotics
Vol. 84, H. Fujita and P. Johannesson (Eds.), New Trends in Software Methodologies, Tools and Techniques
Vol. 83, V. Loia (Ed.), Soft Computing Agents
Vol. 82, E. Damiani et al. (Eds.), Knowledge-Based Intelligent Information Engineering Systems and Allied Technologies
Vol. 81, In production
Vol. 80, T. Welzer et al. (Eds.), Knowledge-based Software Engineering
Vol. 79, H. Motoda (Ed.), Active Mining
Vol. 78, T. Vidal and P. Liberatore (Eds.), STAIRS 2002
Vol. 77, F. van Harmelen (Ed.), ECAI 2002
Vol. 76, P. Sinčák et al. (Eds.), Intelligent Technologies – Theory and Applications
Vol. 75, I.F. Cruz et al. (Eds.), The Emerging Semantic Web
Vol. 74, M. Blay-Fornarino et al. (Eds.), Cooperative Systems Design
Vol. 73, H. Kangassalo et al. (Eds.), Information Modelling and Knowledge Bases XIII
Vol. 72, A. Namatame et al. (Eds.), Agent-Based Approaches in Economic and Social Complex Systems
Vol. 71, J.M. Abe and J.I. da Silva Filho (Eds.), Logic, Artificial Intelligence and Robotics
Vol. 70, B. Verheij et al. (Eds.), Legal Knowledge and Information Systems
Vol. 69, N. Baba et al. (Eds.), Knowledge-Based Intelligent Information Engineering Systems & Allied Technologies
Vol. 68, J.D. Moore et al. (Eds.), Artificial Intelligence in Education
Vol. 67, H. Jaakkola et al. (Eds.), Information Modelling and Knowledge Bases XII
Vol. 66, H.H. Lund et al. (Eds.), Seventh Scandinavian Conference on Artificial Intelligence
Vol. 65, In production
Vol. 64, J. Breuker et al. (Eds.), Legal Knowledge and Information Systems
Vol. 63, I. Gent et al. (Eds.), SAT2000
Vol. 62, T. Hruška and M. Hashimoto (Eds.), Knowledge-Based Software Engineering
Vol. 61, E. Kawaguchi et al. (Eds.), Information Modelling and Knowledge Bases XI
Vol. 60, P. Hoffman and D. Lemke (Eds.), Teaching and Learning in a Network World
Vol. 59, M. Mohammadian (Ed.), Advances in Intelligent Systems: Theory and Applications
Vol. 58, R. Dieng et al. (Eds.), Designing Cooperative Systems
Vol. 57, M. Mohammadian (Ed.), New Frontiers in Computational Intelligence and its Applications
Vol. 56, M.I. Torres and A. Sanfeliu (Eds.), Pattern Recognition and Applications

ISSN: 0922-6389

Real World Semantic Web Applications

Edited by

Vipul Kashyap
National Library of Medicine

and

Leon Shklar
Dow Jones Online

IOS Press
Ohmsha

Amsterdam • Berlin • Oxford • Tokyo • Washington, DC

© 2002, The authors mentioned in the Table of Contents

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, without prior written permission from the publisher.

ISBN 1 58603 306 9 (IOS Press)
ISBN 4 274 90556 X C3055 (Ohmsha)
Library of Congress Control Number: 2002113949

Publisher
IOS Press
Nieuwe Hemweg 6B
1013 BG Amsterdam
The Netherlands
fax: +31 20 620 3419
e-mail: order@iospress.nl

Distributor in the UK and Ireland
IOS Press/Lavis Marketing
73 Lime Walk
Headington
Oxford OX3 7AD
England
fax: +44 1865 75 0079

Distributor in the USA and Canada
IOS Press, Inc.
5795-G Burke Centre Parkway
Burke, VA 22015
USA
fax: +1 703 323 3668
e-mail: iosbooks@iospress.com

Distributor in Germany, Austria and Switzerland
IOS Press/LSL.de
Gerichtsweg 28
D-04103 Leipzig
Germany
fax: +49 341 995 4255

Distributor in Japan
Ohmsha, Ltd.
3-1 Kanda Nishiki-cho
Chiyoda-ku, Tokyo 101-8460
Japan
fax: +81 3 3233 2426

LEGAL NOTICE
The publisher is not responsible for the use which might be made of the following information.

PRINTED IN THE NETHERLANDS

Preface

The Semantic Web is the key initiative being promoted by the World Wide Web Consortium (W3C). Its goal is to provide the next generation Internet infrastructure, where information is given a well defined meaning, better enabling people and programs to work in cooperation with each other. A crucial component of the semantic web is the ability of machines to associate data with meanings. The Resource Description Framework (RDF) is the World Wide Web Consortium's specification for defining machine-understandable metadata. In the long term, we expect that RDF in conjunction with other standards such as Web Services Description Language (WSDL), Simple Object Access Protocol (SOAP), Universal Description Discovery and Integration (UDDI), and XML will serve as the foundation for Semantic Web applications.

In this book, we assembled the best papers that were presented at the Workshop on Real-World RDF and Semantic Web Applications, held in the context of the 11th International World Wide Conference in Hawaii in May 2002, and some invited papers. The title of the workshop is in many ways indicative of the evolution of Semantic Web concepts over the last few years. Early on, at the time of the initial introduction of RDF, Semantic Web was perceived simply as a collection of RDF applications. New specifications, including the DARPA Agent Markup Language (DAML) and the Ontology Inference Layer (OIL), were introduced to strengthen the foundation of the Semantic Web. Since these specifications were defined as RDF applications, their introduction further reinforced the critical importance of RDF.

However, as it became increasingly clear that it is going to be some time until DAML and OIL applications can be considered practical, more and more people started taking a wider view of the Semantic Web. Our initial title for the Workshop was "Real World RDF Applications". As we started on the process of soliciting contributions and communicating with our peers, it became more and more obvious that the only way to build a real world RDF application is to involve other existing standards and technologies. From there, it was a logical step to ask ourselves why not to consider all contributions that serve the long-term goal of building the infrastructure of machine-understandable metadata. In the end, we were happy to have taken this wider approach because it led to interesting if somewhat controversial discussions and the cross-pollination of ideas.

Much of the attention was paid to the possibility of "supercharging" existing standards by using them in conjunction with RDF models for building advanced Web services. The importance of early commercial implementations is that they help to gain insight into the evolution of the Semantic Web technology. Applications that make use of machine-understandable metadata range from information retrieval to Web-enabling of legacy applications. Current developments include metadata-based Enterprise Application Integration (EAI) systems, data modeling solutions, and wireless applications.

A number of articles in this book concentrate on solving general problems of metadata extraction, ontologies and various other aspects of knowledge representation. We tried to select papers that address practical problems that are important for Internet applications, or serve to evolve languages that have strategic importance for the Semantic Web (e.g., RDF, DAML and OIL, etc.).

Examples of the former include the paper "Word Sense Disambiguation and Measuring Similarity of Text Using WordNet" by P. Bhattacharyya and N. Unny, which applies semantic modeling to information retrieval, and the paper "Semantic Enhancement

Engine" by B. Hammond, A. Sheth and K. Kochut, which addresses issues of metadata retrieval and its application to representing collections of heterogeneous distributed documents. Yet another example is the paper by M. Dumas, L. Aldred and A. ter Hofstede, which proposes a mapping between conceptual models and Web forms. Finally, C. Behrens and V. Kashyap address the problem of creating ontologies through the automated analysis of expert opinions.

Examples of the latter include the paper "Change Detection of RDF Documents using Signatures" by L. Khan that addresses the problem of maintaining integrity of RDF specifications. In their paper "Verifying Constraints on Web Service Compositions", Z. Cheng, M. Singh and M. Vouk address the important subject of the representation and composition of Web services. They provide an interesting analysis of WSDL and its limitations. C. Goble and D. De Roure present in their chapter, an interesting discussion and innovative synergy of two strategic technology directions that are in the process of being standardized, namely, Grid Computing and Semantic Web.

Another set of papers is dedicated to the use of semantics modeling and the Semantic Web technologies for specialized applications. One example is the paper by T. Rindflesch and A. Aronson "Semantic Processing for Enhanced Access to Biomedical Knowledge", which addresses domain-specific challenges of extraction of semantic information from biomedical data. Other examples include the paper by P. Assini, who discusses applying semantic modeling to the representation of statistical data, and the paper by J. Ernst, which is dedicated to decision making and knowledge management.

Machine-understandable metadata is emerging as a new foundation for component-based approaches to application development. Within the context of reusable distributed components, Web services represent the latest architectural advancement. RDF Servers enable these two concepts to be synthesized, providing powerful new mechanisms for quickly modeling, creating and deploying complex applications that readily adapt to real world needs. At the same time, we believe that even applications that do not make use of RDF, but are trying to achieve similar goals by using other specifications such as XML, WSDL, UDDI, SOAP and XSL, create fertile grounds for the future. In the short term we expect RDF to be used in conjunction with the above standards, though new advancements may see re-alignments and merging of the various standards.

Vipul Kashyap,
National Library of Medicine

Leon Shklar,
Dow Jones Online

Foreword

The Semantic Web is a vision of the next generation of the World Wide Web, based on the idea of associating *formal semantics* with on-line content. On the Semantic Web, automated agents will be *acting on behalf* of users, relieving them of the burden of a multitude of menial tasks that now are too difficult to automate. The Semantic Web represents an opportunity to elevate the level of *interoperability* between applications and systems, since they could now exchange information they were not strictly designed to process.

Early on – during the initial development of the RDF standard – we envisioned the Semantic Web to represent a true *chicken-and-egg* problem: until there is a sufficient amount of semantic data available, there is little incentive to produce software to process it, and without software for semantic data, where is the incentive to produce the data? Fortunately, more and more developers are challenging this and have embarked on a mission to make the Semantic Web a reality. This book is about real-world applications employing ontological and "semantic" techniques and technologies, and some of the "pioneer" developers spearheading the Semantic Web revolution are represented.

The Semantic Web is a team sport, and it is not enough for just one organization to play: unlike some other new technologies, it cannot be deployed by one organization or via one application alone. Being largely about interoperability, the true benefits of the Semantic Web are only realized if a certain "critical mass" is reached. It is all about sharing.

Cynically, one may ask: What is the financial benefit of interoperability? Currently, there is no direct revenue from it, and in some cases making things more amenable to automation will make it more difficult to generate revenue. For example, if a content provider relies on advertisement revenue, there may not be any if only agents visit his web site. Semantic representation of on-line content is also predicated on the existence and sharing of ontologies, and there is only an indirect incentive for someone to share an ontology (i.e., the promise of elevated interoperability). Hopefully, the obvious long-term benefits of the advent of the Semantic Web will be incentive enough to motivate developers and content providers to overcome the short-term deployment obstacles. Clearly, once everyone plays, the Semantic Web will be an indispensable thing: it will become more expensive not to play, just as it is currently risky for a company not to have any web presence.

It is my dream that an "ecosystem" for the Semantic Web will form, lowering the threshold for entry into this game. Part of the ramp-up of this ecosystem is the invention and emergence of new *sustainable business models*, ones that will encourage the creation and sharing of semantic information. Ultimately, it may well be that the greatest challenges for this wonderful technology are not technological after all, but have more to do with the business aspects. Naturally, this is not to say that there are no technological challenges as well.

Ora Lassila,
Chief Scientist, Nokia Venture Partners
and editor of the W3C RDF Model and Syntax Specification

This page intentionally left blank

Contents

Preface, *V. Kashyap and L. Sklar* v
Foreword, *O. Lassila* vii

Metadata, Ontologies and Knowledge Representation
Word Sense Disambiguation and Measuring Similarity of Text Using WordNet, *P. Bhattacharyya and N. Unny* 3
Semantic Enhancement Engine: A Modular Document Enhancement Platform for Semantic Applications over Heterogeneous Content, *B. Hammond, A. Sheth and K. Kochut* 29
From Conceptual Models to Constrained Web Forms, *M. Dumas, L. Aldred and A.H.M. ter Hofstede* 50
The "Emergent" Semantic Web: A Consensus Approach for Deriving Semantic Knowledge on the Web, *C. Behrens and V. Kashyap* 69

Internet Applications and Standards
Change Detection of RDF Documents Using Signatures, *L. Khan, L. Wang and Q. Chen* 93
Verifying Constraints on Web Service Compositions, *Z. Cheng, M.P. Singh and M.A. Vouk* 117
Semantic Web and Grid Computing, *C. Goble and D. De Roure* 131

Semantic Web and Specialized Applications
Semantic Processing for Enhanced Access to Biomedical Knowledge, *T.C. Rindflesch and A.R. Aronson* 157
NESSTAR: A Semantic Web Application for Statistical Data and Metadata, *P. Assini* 173
Collaborative Decision Making and Personal Knowledge Management with R-Objects Pepper™, *J. Ernst* 184

Author Index 195

This page intentionally left blank

Metadata, Ontologies and Knowledge Representation

This page intentionally left blank

Real World Semantic Web Applications
V. Kashyap and L. Shklar (Eds.)
IOS Press, 2002

Word Sense Disambiguation and Measuring Similarity of Text Using WordNet

Pushpak Bhattacharyya and Narayan Unny
Department of Computer Science and Engineering,
Indian Institute of Technology,
Bombay 400 076, INDIA.
{pb,nue}@cse.iitb.ernet.in

Abstract. This paper advocates the use of lexical knowledge and semantics to improve the accuracy of information retrieval. A system of measuring text similarity is developed, which attempts to integrate the meaning of texts into the similarity measure. The work hinges on the organization of the synsets in the WordNet according to the semantic relations of *hypernymy/hyponymy*, *meronymy/holonymy* and *antonymy*. A unique measure using the *link distance* between the words in a *subgraph of the WordNet* has been evolved. This needs word sense disambiguation which in itself is a complex problem. We have developed an algorithm for: word sense disambiguation exploiting again the structure of the WordNet. The results support our intuition that including semantics in the measurement of similarity has great promise.

1 Introduction

Today's search engines like *google* do an admirable job of information retrieval. However, the goal of *retrieving all and only the most relevant information* is still a far cry. One step towards realizing this ideal goal is the *detection of similarity of texts, i.e., judging how close in meaning two given texts are*. Fundamentally, the similarity of two objects is measured by the number of features the objects have in common. This idea is applied to text similarity detection also. However, approaches differ on the notion of what the features of a text object are. In contempoary IR, the words of a text are taken as features. The more features the two texts share, the more similar they are to each other.

1.1 Inclusion of meaning

One obvious shortcoming of the *bag of words* approach is that it does not at all consider the meaning of texts. Two phenomena that need to be considered in this context are *polysemy* and *synonymy*. *Polysemy* refers to the same word form having different meanings in different contexts, while *synonymy* refers to different word forms having the same meaning. Consider, for example, two texts each using the term *board* extensively. But in one, the term means *wood, plank, etc.* and in the other it means *committee*. Failure to detect this polysemy leads to *overextimating the similarity value*. On the other hand, consider two texts, one using the term *wood* and the other using *plank*. The similarity measure not accounting for the synomymy here leads to *underestimating the similarity value*.

These problems with the conventional approach necessitate considering the semantics of the text. The idea is to *expand a term so that it covers not only its synonyms but also the words which are closely related to it. This is the basic intuition behind our algorithm and is implemented using the WordNet.*

This article is organized as follows. In section 2 we discuss the WordNet. Section 3 describes the common similarity measures. In section 4 we give the block diagram of the system and describe the top level design. The word sense disambiguation module along with its evaluation is discussed in section 5. The description of the actual similarity estimation is given in section 6. Detailed evaluation of the system is given in section 7. We conclude in section 8.

2 WordNet

The lexical resource WordNet [2] plays the central role in the work described in this article. English nouns, verbs, adjectives and adverbs are organized into synonym sets, each representing one underlying lexical concept. Different semantic relations link these synonym sets.

As discussed in the previous section, synonymy and polysemy make it impossible to have a one to one mapping of word forms to the meanings. This was the inspiration behind the organization of WordNet into *synsets*. A synset is a set of synonymous words whose primary function is to capture a unique meaning. For example, the word *board* can have two different meanings, *viz.* a *piece of lumber* and *a group of people*. The synonym sets, {*board, plank*} and {*board, committee*} can serve as unambiguous designators of these two meanings of *board*. These synonym sets do not explain what the concepts are; they merely signify that these unique concepts exist.

The most important relation for WordNet is **synonymy** which denotes the similarity of meaning, since the ability to judge that relation between word forms is a prerequisite for the representation of meanings. The definition of synonymy can be given as follows: *two expressions are synonymous in a linguistic context* **C** *if the substitution of one for the other in* **C** *does not alter the truth value*. For example, the substitution of *plank* for *board* will seldom alter truth values in carpentry contexts.

The next familiar relation is **antonymy** which for a given word is a word with opposite meaning. Antonymy provides a central organizing principle for the adjectives and adverbs in WordNet. One important observation is that the words from the same synset have definite preference for words from the synset with opposite meaning. For example, *rise* takes *fall* as the antonym and *ascend* takes *descend* as the antonym even though {*rise, ascend*} and {*fall, descend*} are two synsts.

Unlike synonymy and antonymy, which are lexical relations between word forms, **hypernymy/hyponymy** is a semantic relation between word concepts, *i.e.*, synsets. This semantic relation arranges the synsets in a hierarchy. For example, {*dust, debris, junk, rubble*} is a hyponym of {*rubbish, trash*} which in turn is a hyponym of {*substance, matter*}. Much attention has been devoted to hyponymy/hypernymy (variously called subordination/superordination, subset/superset, or the IS-A relation). Hyponymy is transitive and asymmetrical, and since there is normally a single superordinate, it generates a hierarchical semantic structure in which a hyponym is said to be below its superordinate.

The semantic relation *part-whole* or *HAS-A* is known to as **meronymy/holonymy**. The meronymic relation is transitive (with qualifications) and asymmetrical, and can be used to construct a part hierarchy (with some reservations, since a meronym can have many holonyms).

For example, {*house*} has as meronym {*study*} which in turn has as meronym {*door*}.

The semantic relations represent associations that form a complex network. Knowing where a word is situated in that network is an important part of knowing the word's meaning. This network of synsets forms the WordNet.

3 Similarity measures

In this section, we look at some of the similarity measures that have been used extensively in the field of information retrieval. These measures are discussed in two parts: (1) Text similarity measures [12] and (2) Conceptual similarity measures [10]. In this article we have studied the use of the latter for solving the former problem.

3.1 Text Similarity

Text similarity measures essentially take the bag-of-words representation of text for measuring similarity. Discussion on some of these measures follows:

1. **Cosine**
   This is the most widely used measure. The popularity stems from its simplicity. It is calculated from the cosine of the vectors corresponding to the two texts being compared. The vector of a text is formed by using the frequency of occurrence of distinct words in the text as the components. Thus this measure gives the intersection of the two texts weighted by the respective frequencies of occurrence. Mathematically, it is denoted as,

$$Cos(d,q) = \frac{\sum_{(t\in q)\wedge(t\in d)} f_{d,t} f_{q,t}}{\sqrt{(\sum_{t\in q} f_{q,t}^2)(\sum_{t\in q} f_{d,t}^2)}}$$

   where $f_{x,t}$ is the frequency of term t in document x.

2. **Dice**
   The dice coefficient is defined by the binary model of documents. Here the components of the document vector are binary values corresponding to the occurrence (or non occurrence) of the term in the document. If X and Y are the documents we need to compare, then the *Dice Similarity* is defined as:

$$Dice(X,Y) = \frac{2 \mid X \cap Y \mid}{\mid X \mid + \mid Y \mid}$$

   where,

   $\mid X \cap Y \mid$ is the number of words that are common in the documents X & Y

   $\mid X \mid$ is the number of terms in document X

   $\mid Y \mid$ is the number of terms in document Y

3. **Jaccard**
   This similarity measure is a slight variant of the dice coefficient and is also based on the

commonality between the two documents that we want to compare. The *Jaccard Coefficient* for two documents X and Y is given by:

$$Jaccard(X, Y) = \frac{| X \cap Y |}{| X \cup Y |}$$

This definition of the coefficient gives the ratio of the common terms to the total number of terms between the two documents X and Y. This formulation can be adapted to the frequency model of the document as well as the binary one.

3.2 Conceptual Similarity

Given a hierarchical graph of IS_A relation among concepts, it is observed that *the similarity between two concepts in this has inverse relationship with the distance between them*, and this distance is termed as the *conceptual distance*. There are two basic appraoches based on this theme.

1. **Information Theoretic**
   There are a number of algorithms which apply the principle of conceptual similarity to the WordNet graphs. [11] uses a measure based on the information content that the two concepts share. The basic intuition behind this approach is that the more the information the two concepts share, the more similar they tend to be. The information shared by two concepts is proportional to the information content of the concepts that subsume them in the taxonomy. Formally, define

   $$sim(c_1, c_2) = max_{c \in S(c_1, c_2)}[-\log p(c)]$$

   where $S(c_1, c_2)$ is the set of concepts that subsume both c_1 and c_2. Here $p(c)$ is the probability of encountering an instance of concept c. Notice that although similarity is computed by considering all upper bounds for the two concepts, the information measure has the effect of identifying minimal upper bounds, since no class is less informative than its superordinates. For example, in WordNet, *NICKEL* and *DIME* are both subsumed by *COIN*, whereas the most specific superclass that *NICKEL* and *CREDIT CARD* share is *MEDIUM OF EXCHANGE* (as shown in figure 1). Note that this approach gives only the similarity between two concepts and not between the two texts.

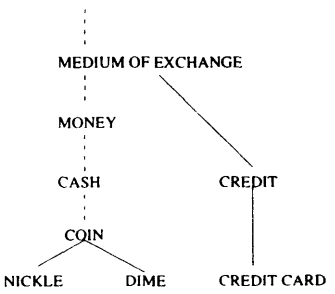


Figure 1: Figure showing the hierarchical relation in the WordNet

2. **Link based**
   The simple measure of link distance between two concepts in the WordNet as an estimate of the similarity between them has been used extensively [6]. In this approach, the number of relational links separating the two synsets is taken as a inverse measure of the similarity between them. For example, the distance between the synsets {*car, auto*} and {*vehicle*} is 2 in the WordNet graph, while the distance between {*car,auto*} and {*table (as a furniture)*} is 7. This means that the former pair of synsets are more similar in meaning than the latter.

4 Our system

The block diagram showing the design of our system is given in figure 2. The system has

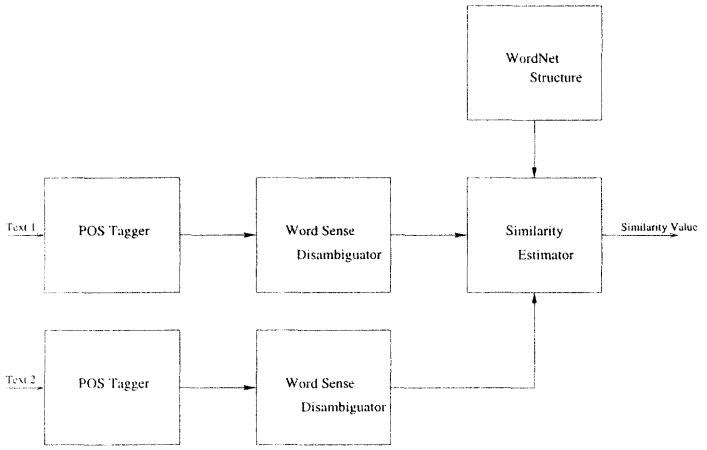


Figure 2: Block diagram of the system

4 independent modules – *Part-Of-Speech (POS) tagger*, *word sense disambiguator*, *WordNet* and *similarity estimator*. The system works by first extracting the nouns from the text using a part-of-speech tagger which in this case is the java based *Qtag*[7] *(vide* tables 1, 2 and 3 for two sample texts, the corresponding tagged outputs and the nouns extracted there from respectively). These nouns are then fed into the next module – the word sense disambiguator which maps the nouns to the corresponding synsets in the WordNet. These are then passed to the similarity estimator module which uses the WordNet graph and *link distance* to obtain the similarity value.

5 Word Sense Disambiguator module

The first landmark attempt at word sense disambiguation [14] used a supervised approach based on *Roger's Thesaurus*. It worked on a training corpus to learn the correspondence between the sense of a word and the context in which it occurs. After that, the research on word sense disambiguation received a boost with the development of the WordNet which provided a complex and well organised database of words and its senses. The fundamental Idea behind the use of the WordNet for WSD is first described.

There are aesthetic and recreational reasons for an interest in fishes. Millions of people keep live fishes in home aquariums for the simple pleasure of observing the beauty and behaviour of animals otherwise unfamiliar to them. Sportfishing is another way of enjoying the natural environment, also indulged in by millions of people every year. Interest in aquarium fishes and sportfishing support multimillion - dollar industries throughout the world.	Fishes are of interest to humans for many reasons, the most important being their relationship with and dependence on the environment. A more obvious reason for interest in fishes is their role as a moderate but important part of the world 's food supply. This resource, once thought unlimited, is now realized to be finite and in delicate balance with the biological, chemical, and physical factors of the aquatic environment. Overfishing, pollution, and alteration of the environment are the chief enemies of proper fisheries management, both in fresh waters and in the ocean.

Table 1: Sample texts T_1 and T_2

There/EX are/BER aesthetic/JJ and/CC recreational/JJ reasons/NNS for/IN an/DT interest/NN in/IN fishes/NNS ./. . . .	Fishes/NNS are/BER of/IN interest/NN to/TO humans/NNS for/IN many/DT reasons/NNS ,/, the/DT most/RBS important/JJ being/BEG their/PP$ relationship/NN with/IN and/CC dependence/NN on/IN the/DT environment/NN ./. . . .

Table 2: POS tagged words from texts T_1 and T_2

5.1 WordNet based word sense disambiguation

Texts represent flow of ideas through structured sentences. The words which convey the main ideas are commonly known as *keywords*. A common hypothesis in all WSD work is that *although all the words are necessary in a text, it is the nouns the carry the main burden of the expression*. Also important is the observation that given the topic of a text, there is a high probability that most of the words in the text are very closely *related* to the words used for describing the topic. For instance, in a text about *car* we expect to find a lot of words related to *car*, like *steering*, *wheels* etc. This implies that **most of the synsets corresponding to the words in a text lie close to each other when mapped on to the WordNet graph**. This type of constraint imposed on the senses of a group of contiguous words to disambiguate them has been used in many algorithms. The first such algorithm was by Michael Sussna [13] which takes a group of contiguous words in a window, and the sense combination for the words in the window is chosen such that the sum of distances between the synsets corresponding to the word forms and word sense combinations is minimized. Another algorithm that considers this proximity of synsets is the one by Eneko Agirre and German Rigau [1]. In this, they define a *measure of conceptual density* as the number of words in the context window that lie in the sub hierarchy of synsets of the candidate word. The synset of a word is chosen such that the conceptual density is maximized.

The main drawback of these algorithms is that they assume the words lying close to each other in the text to have similar senses. This might not be necessarily true. It has been observed that the reference of a word can extend up to a large distance in the text. Use of anaphors is a good example in point. Hence, though we can be sure that the text is popu-

lated with similar words pertaining to the main topic of the text, we cannot make any such assumption about the *contiguous words* in the text. The proof of such a conclusion lies in the result obtained in [1]. In the evaluation of their algorithm it was seen that the precision of disambiguation increased with the increase in the window size used for disambiguation. This indicates that the constraint of proximity in the WordNet graph works better as the context size increases.

reasons, interest, fishes, Millions, people, fishes, aquariums, pleasure, beauty, behaviour, animals, way, environment, millions, people, year, Interest, aquarium, fishes, support, multimillion, dollar, industries, world.	Fishes, interest, humans, reasons, relationship, dependence, environment, reason, interest, fishes, role, part, world, food, supply, resource, balance, factors, environment, pollution, alteration, environment, enemies, fisheries, management, waters, ocean.

Table 3: Nouns from texts T_1 and T_2

Another flaw in the above description of a text structure is that a text can have more than one theme. This factor must be taken into consideration when designing the WSD algorithms. The idea of capturing the flow of ideas in the text has been previously used in lexical chains [9][4]. The aim in [4] is automatic hyperlinking. Our work has been inspired by lexical chain like structures.

The discussion so far can be summarized by an illustration of how the words in a text can represent the ideas when they are mapped onto the synsets of the WordNet graph.

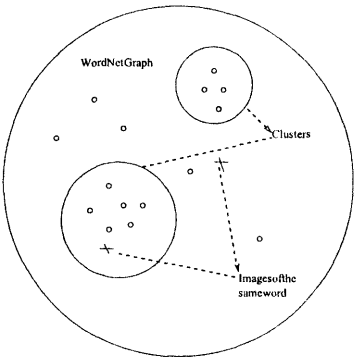


Figure 3: Mapping of words in a text onto WordNet

In figure 3, the large circle represents the WordNet graph. The smaller circles inside them correspond to the clusters formed by putting together similar words in the text. The points within the smaller circles are the synsets corresponding to the words in the text. Each cluster stands for an idea in the text.

When a particular word is not disambiguated there is more than one cluster for the word in the WordNet graph. If our previous assumption of clustering- the fact that *words in the text are within close semantic distance from one another*- is true, then we should be able to choose the sense of the word in such a way that we get a closely packed cluster. The question now is - where do we start from to obtain the clusters in the given text?

5.2 Our approach

We start building the clusters from the Monosems which are words having a single sense. Thereafter, the algorithm works by building a directed acyclic graph (DAG) over the words in the text and using the properties of the *conceptual distance*. We call the DAG the *semantic graph*. In this *semantic graph* the nodes are synsets of WordNet and the links represents the fact that there is a path in the WordNet, between the two synsets within the *search_depth* radius. The steps of the algorithm are now enumerated:

1. **Collect the Monosems:** Monosems form the roots of the semantic DAG that gets built as the WSD algorithm proceeds. For a single word in the text we have all the senses of the word as nodes in the DAG. The number of roots of the DAG becomes equal to the number of monosems in the text.

2. **Initialize the scores of the DAG:** For each node in the DAG, it has a score corresponding to the probability of the word taking that sense. To start with, the roots of the DAG are monosems. These are initialized with a score of 1.

3. **Find the link distance to other words:** Each node in the DAG is the synset corresponding to a word in the text. We take a word from the text that has not been added to the DAG yet and for all synsets corresponding to the word, we find the link distance between the synsets corresponding to the current node of the DAG and the synsets of the selected word from the text. This involves searching the WordNet graph starting from one of the synsets and proceeding in a breadth first fashion till we reach the other synset. As for the links, all kinds of relation links in the WordNet are considered.

   To decrease the time and space complexity of a breadth first search we restrict the search to a cut-off radius which we call **search_depth**. The value of this variable decides the depth upto which the search proceeds. Any word at a link distance more than the *search_depth* is ignored. It is easy to see that the more the *search_depth*, the more is the precision of disambiguation. The *Search_depth* also controls the value of the similarity assigned to pairs of synsets.

4. **Form the semantic DAG:** After finding the distance to the synsets of a word we compare the distances that we have obtained. The word along with all its senses to which a path has been found is added as the child of the DAG node. Two situations arise in this case:

   (a) The new DAG node is already present: This condition arises when the combination of the word and the sense was previously found by a sibling of the current node. In such a case an addtional parent pointer for the existing node is created.

   (b) The new DAG node is not present: In this case the node is created and added as a child of the current DAG node.

   The frontier of the DAG is expanded one level at a time in accordance with the breadth first search. After a particular level has been added to the DAG, it goes onto expanding the next level. When a DAG node is chosen as the current node, the word corresponding to the current selection is permanently removed from the list of words in the text. Therefore *a node corresponding to a particular word at a particular position in the text can occur only once in the DAG*. This is done to avoid cycles. Note that, this does not prevent the

same word occurring at different points in the text from appearing more than once in the DAG at different levels.

5. **Pass the score of the parent to the child:** Having initialized the scores of the roots to 1, we need to assign scores to the subsequent nodes at each level. Let us denote a node at level i and sense j as W_j^i. The score of W_j^i depends on the following three parameters.

   (a) *Score of parent*: Since a DAG node is added based on its proximity to the synset of the corresponding parent DAG node, the probability of the node calculated as

   $$Score(W_j^i) \propto Score(W_j^{i-1})$$

   (b) *Distance between the child and the parent*: The score of a new DAG node also depends on its distance from the parent node. The intuition behind this is that similar words are distributed quite close to each other. This phenomenon is evident in texts having multiple paragraphs. Mostly, the topic of discussion changes during the transition from one paragraph to another. Similar words can be found inside a paragraph rather than between paragraphs. Therefore,

   $$Score(W_j^i) \propto \frac{1}{Dist(W^i, W^{i-1})}$$

   (c) *Link distance in the WordNet*: The score of a word also depends on the link distance of the parent word from the current word in the DAG. The lesser the distance between them, the more similar the child node is to the parent node. We are then more certain about the sense of the child node. The score then is

   $$Score(W_j^i) \propto \frac{1}{Link\_dist(W^i, W^{i-1})}$$

   where *Link_dist* is the numerical value of the distance between the parent and child nodes in terms of the number of links separating them in the WordNet graph. If this distance is larger than the parameter *search_depth* then the *link_dist* is taken as infinite.

   Finally combining,

   $$Score(W_j^i) = \frac{Score(W_k^{i-1})}{Dist(W_j^i, W_k^{i-1}) \times Link\_dist(W_j^i, W_k^{i-1})}$$

6. **Judge the sense for a particular word:** Having assigned the scores to all the nodes of the DAG, we compare the scores of all the senses of a word. The sense with the highest score selected. *Since we remove the word from the text after having added it to the DAG, all the senses of a particular word occur at the same level of the DAG.*

The DAG produced by the above algorithm is a subgraph of the WordNet. This subgraph contains most of the words of the given text. A property of the created DAG is that the accuracy of disambiguation decreases with the depth of the nodes. This is because of the *cascading effect of errors* in the disambiguation process. To start with, one is sure about the sense of the words which are monosems. But as the score is passed from one level to another, the certainty about the proportionality between the score and the senses decreases. This means that if an error occurs at a level and a wrong sense of a word gets a high score, then this error can percolate down.

One technique to improve the precision is to maintain a cut-off at some level of the DAG. All the levels below this level are excluded from the disambiguation process.

5.3 An Example to illustrate the Algorithm

We demonstrate the steps of the algorithm using an example of a document from the Semcor corpus shown below:

Nevertheless, "we feel that in the future Fulton_County should receive some portion of these available funds", the jurors said. "Failure to do this will continue to place a disproportionate burden" on Fulton taxpayers. The jury also commented on the Fulton ordinary's court which has been under fire for its practices in the appointment of appraisers, guardians and administrators and the awarding of fees and compensation.

In the above, the underlined words are the nouns in the text and most of them have multiple senses. For example, *portion* has 6 WordNet senses as a noun.

(Step 1) Collect the monosems: Some of the monosems in the given text- as given in the WordNet- are *funds*, *jurors*, *guardians*, *awarding*, and *taxpayers*.

(Step 2) Initialize the scores of monosems: The words *funds*, *jurors*, *guardians*, *awarding* and *taxpayers* form the roots of the DAG and have a score of 1 each.

(Step 3) Find the link distance to other words: From each leaf of the DAG, the link distance is found to the words in the text. For example, a link distance of 2 is found from the sense no. 1 of *funds*- the only sense- to the sense no. 4 of *portion*. This is added as a child of *funds*.

(Step 4) Form the semantic DAG: Step 4 is repeatedly performed to grow the DAG structure. A part of the DAG structure is shown in figure 4.

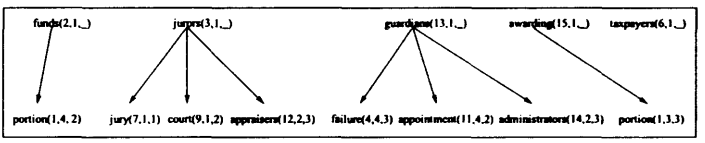


Figure 4: A partial DAG structure

In figure 4, monosems occupy the roots of the DAG. The three numbers given in brackets along with the words represent *the position of the word in the text, the sense number* and *the link distance to that sense from the parent node*, respectively.

(Step 5) Pass the score of the parent to the child: The nodes of the DAG are then assigned scores based on the relation,

$$Score(W_j^i) = \sum \frac{Score(W_k^{i-1})}{Dist(W_j^i, W_k^{i-1}) \times Link\_dist(W_j^i, W_k^{i-1})}$$

where the summation is over all the parents W^{i-1} of W^i. The DAG structure with the scores assigned to the nodes is shown in figure 5.

Figure 5 illustrates the final structure of the DAG that has been built. The numbers in brackets assigned to the nodes in the DAG represent the index of the word in the text, the

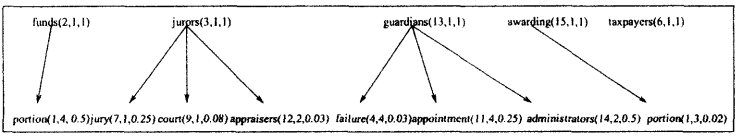


Figure 5: A partial DAG structure with scores

sense number and the score of that particular node. For example, for the node *portion* the index is 1, the sense no. is 4 and the score is 0.5.

(Step 6) Judge the sense for a particular word: The sense for a word is the one with the highest score. For example, in figure 5, the 4th sense of the word *portion* with the score 0.5 is chosen. Though the actual sense of *portion* in the text is 3, sense 4 too is quite close. Thus the algorithm narrowed down the choice from the six senses to two in this case.

The other good example of disambiguation is *court* which has 9 senses.

5.4 Evaluation of the sense disambiguator module

The evaluation was done using three different parameters, *viz.*, **precision**[1], **recall**[2], and **coverage**[3]. For our experiments, we chose documents from the Semcor corpus [8]. This is a corpus in which the words have been tagged with their sense number in the wordnet, the part of speech and other lexical information.

In our experiment, we chose the Semcor corpus document **br-a01**. This text has been used by other WSD algorithms also [6]. Nouns extracted from this text and stripped of any sense tagging are passed to the proposed algorithm as input. The output is the same set of nouns with the WordNet sense marking. The senses obtained are compared with the actual senses of these nouns. The performance is then computed from the counts we obtain. We performed the experiment for three different values of the search_depth. The results are summarized below.

Search Depth	Precision	Recall	Coverage
3	62.87	32.30	51.38
4	57.50	42.46	73.84
5	51.91	45.84	88.30

Table 4: Results of the experiment

From the results we find a neat variation in the precision and recall values with the values of the search_depth parameter. This variation of precision can be explained by the fact that the search_depth restricts the distance upto which two synsets can be considered to be similar. If

[1]Precision is defined as the ratio of the number of words disambiguated correctly, to the number of words disambiguated in total.

[2]Recall is defined as the ratio of number of words *correctly* disambiguated to the total number of words that were input to the algorithm.

[3]Coverage is defined as the ratio of number of disambiguated words to the total number of words that were input to the algorithm.

we increase the value of this parameter, even synsets that are far apart are considered similar, resulting in a dilution of the similarity measure. This in turn affects the precision of the disambiguation.

The results of our experiments compared with the results of the algorithm using *conceptual density* [6] have been summarized in table 5.

Algorithm	Precision	Recall	Coverage
Conceptual density	47.30	39.40	83.20
Semantic graph	62.87	32.31	51.38

Table 5: Comparison of performance

Considering the running example of the two texts in table 1, the sense disambiguation module gets the list of nouns in table 3. These are then disambiguated by our WSD module. The list of disambiguated words with their WordNet sense numbers are shown in table 6. Here, the value of *search_depth* was set to 4.

fishes(1), Millions(1), people(1), fishes(1), aquariums(1), animals(1), millions(1), people(1), aquarium(1), fishes(1), world(1), world(1)	Fishes(3), interest(7), humans(1), interest(7), fishes(2), part(4), world(5), food(1), enemies(3), fisheries(1), waters(1), ocean(1), ocean(1)

Table 6: List of disambiguated nouns

6 Similarity Estimation

We use the conceptual distance between constituent words to estimate the similarity between texts. Constructing a similarity measure between two documents from the similarity of words is a unique feature of our system. The output of the system is a similarity value in percentage.

Steps of the algorithm

The algorithm has three important steps:

- Get the list of synsets in each of the texts.
- For each synset S_1 in $text_1$ do,
  For each synset S_2 in $text_2$ do,
  For i = 0 to MAX do,
  Find all the synsets that are at a distance of i
  relational links from S_1in the WordNet graph.
  If S_2 is among them then increment count in L_i.
- Compute the similarity using the relations shown below.

L_i is number of links obtained between the synset of two documents at the distance of i. MAX is the maximum specified radius of search on the WordNet graph. This means that the

search for a synset in the neighbourhood of another synset is restricted only to a link distance of MAX. We define the parameter *Total_Weighted_links* as,

$$Total\_Weighted\_links = \sum_{i=0}^{MAX-1} L_i \times [MAX - i]$$

Here, we compute the total number of links obtained at various radii of search on the WordNet graph, by weighting the number of links by the proximity of distance. This means that the links at a shorter distance gets more weightage than the links at larger distances. This is a direct result of the fact that conceptual distance is inversely related to the similarity between concepts.

If N_A and N_B are the number of synsets in the two documents A and B, then the maximum possible links is $N_A \times N_B$. Then the *link_similarity* is defined to be,

$$link\_similarity(A, B) = \frac{Total\_Weighted\_links}{maximum\ number\ of\ links}$$

$$= \frac{\sum_{i=0}^{MAX-1} L_i \times [MAX - i]}{N_A \times N_B}$$

This measure gives just the plain similarity in terms of the links between the two texts A and B. This similarity measure does not satisfy the property *link_similarity(A,A)= 1* because the numerator that stands for the total weighted number of links need not be equal to the denominator which is the maximum possible number of links. Hence, we need to normalise this measure with the self-similarity of the text documents themselves. The final measure of similarity between the two text documents is,

$$semantic\_similarity(A, B) = \frac{link\_similarity(A, B)}{\sqrt{link\_similarity(A, A) \times link\_similarity(B, B)}}$$

It is interesting to note the correspondence between this measure and the dot-product of the cosine similarity.

As is apparent from the above algorithm, the similarity measure between two documents is found from a similarity measure of the constituent words.

6.1 An example

We illustrate the working of this module using the running example of previous sections. The list of disambiguated words given in table 6 is passed as input to this module. It produces as output the number of links at each of the distances less than *MAX*. These are then used to find the similarity between the two texts. Here, the value of *MAX* used was 3.

In the table 7 the left hand side of the arrow corresponds to word from the first text and the right hand side corresponds to those from the second text. The number given in the bracket gives the link distance between the two words. Table 7 gives the links found between text1 and text2. Tables 8 and 9 give the self similarity links for the two texts.

From table 7 we find that there are 3 links at distance 1 and 1 link at distance 2. The number of synsets in text1 and text2 are 12 and 13 respectively. Therefore,

$$link\_similarity(text1, text2) = \frac{3 \times 1 + 2 \times 3 + 1 \times 0}{12 \times 13}$$

people ⟹ human(1)
people ⟹ world(1)
people ⟹ enemy(1)
animal ⟹ human(2)

Table 7: The links between the two texts

million ⟹ million (0)
people ⟹ people(0)
aquarium ⟹ aquarium(0)
animal ⟹ animal(0)

Table 8: The self similarity links for text1

fish ⟹ fish(0)
fish ⟹ food(2)
interest ⟹ interest(0)
human ⟹ human(0)
human ⟹ world(2)
human ⟹enemy(2)
part ⟹ part(0)
world ⟹ human(2)
world ⟹ world(0)
world ⟹ enemy(2)
food ⟹ fish(2)
food ⟹ food(0)
enemy ⟹ human(2)
enemy ⟹ world(2)
enemy ⟹ enemy(0)
fishery ⟹ fishery(0)
water ⟹ water(0)

Table 9: The self similarity links for text2

$$link\_similarity(text1, text2) = 0.057$$

Similarly we compute *link similarity(text1, text1)* and *link similarity(text2, text2)*. These values are then plugged into the formula for *semantic_ similarity* to get its final value as **61.69%**.

7 Evaluation of the System

The efficacy of the similarity measure that we have evolved has been tested against the universally used cosine similarity measure. This is consistent with the practice adopted in the field of text clustering research.

The experiments have been performed in three different settings.

1. The first is the one in which just the basic task of *computing and comparing the similarity measures* is performed. Ninety nine documents from 17 classes are taken and pairwise inter-class similarity is computed using both our measure and the cosine similarity measure.

2. The second setting is *classification*. These 99 documents are split into training and testing sets and are put into classes using both our measure and the cosine similarity measure. The accuracy of the classification on the test set throws light on the effectiveness of the similarity measure.

3. The third setting is *clustering*. The 99 documents are clustered using both our measure and the cosine similarity measure.

In the first setting we wanted to test the *raw effectiveness* of the proposed similarity measure. Since multiple senses of words interfere, we have performed the tests with 100% correct sense tagged texts. We have used the *Semcor* corpus which is a subset of the famous *Brown corpus* tagged with WordNet sense number of the words. We call this the ideal situation.

In the second setting- which is more realistic- the text tagged with only part-of-speech but no sense tagging has been taken through the word sense disambiguation stage followed by classification using the $K - NN$ method. The underlying similarity measures for classification are our measure and the cosine similarity measure. The accuracy of the classification is expected to stand witness to the effectiveness of the similarity measure.

In the third setting the text tagged with only part-of-speech but no sense tagging has been taken through the word sense disambiguation stage followed by clustering. The underlying similarity measures for clustering are our measure and the cosine similarity measure. The accuracy of the clustering is expected to throw light on the effectiveness of the similarity measure.

All the experiments are performed using only the nouns in the documents. The reason- it is reiterated- is that nouns carry the burden of information in the text.

7.1 Corpus description

We have used the documents of the Semcor corpus for the purpose of evaluation. Semcor corpus is a subset of the *Brown Corpus*[3] consisting of texts that have been tagged with POS and the WordNet senses. Semcor consists of about 186 documents classified into 20 classes. We have chosen a subset of the documents and the classes as shown in the table 10.

S.No.	Class Names	Document identification in Brown corpus	Number of documents
0	Political	br-a01- a02,b20,g01,h21,j37-38,j42	8
1	Law	h9,12,16,17	4
2	Taxation	h24	1
3	Science and Technology	e25-26,g11,d22,j01-20,j53	25
4	Science Fiction	m01-02	2
5	Arts	c04,g16,j59	3
6	Fiction(Mystery)	l01-l18	11
7	Fiction(Adventure)	n05,n09-17,n20	10
8	Fiction(Romance)	p01,p07-10,p12,p24	6
9	Fiction(Humour)	r04-09	6
10	Literature	g12,g15,g28,g44	4
11	Psychology	j29,j31	2
12	Management	g20,j30	2
13	Linguistics	j32-35	4
14	History	j54	1
15	Religion	d01-04,g21	5
16	Sports	a11-a15	5

Table 10: Classes and their respective documents

When the experiments are performed with WSD module on, this data is passed on after stripping the sense information.

7.2 Setting-1: Basic comparison of similarity measures

To get the feel of the similarity values of the documents in the corpus, we used the sense tags of the words given in the Semcor corpus. The nouns along with their senses were fed into the similarity estimator module. This corresponds to the *fully correct sense* criterion as discussed earlier. The same nouns were then used to obtain the similarity values as given by the cosine similarity measure.

For the 17 classes described in table 10 the mean and variation of similarity have been shown in tables 11 and 12.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0.32,0.27																
1	0.22,0.08	0.44,0.33															
2	0.23,0.06	0.29,0.04	1.00,0.00														
3	0.13,0.10	0.13,0.10	0.15,0.10	0.20,0.19													
4	0.27,0.15	0.30,0.13	0.40,0.01	0.27,0.19	0.87,0.13												
5	0.16,0.10	0.16,0.08	0.18,0.05	0.16,0.12	0.34,0.15	0.46,0.39											
6	0.21,0.16	0.25,0.16	0.33,0.12	0.22,0.20	0.61,0.24	0.29,0.18	0.60,0.32										
7	0.19,0.15	0.22,0.15	0.29,0.11	0.21,0.19	0.58,0.21	0.27,0.17	0.53,0.29	0.57,0.28									
8	0.21,0.16	0.23,0.16	0.32,0.12	0.21,0.20	0.61,0.23	0.29,0.18	0.56,0.29	0.53,0.26	0.60,0.31								
9	0.22,0.13	0.25,0.14	0.32,0.10	0.22,0.18	0.59,0.21	0.31,0.13	0.53,0.26	0.49,0.25	0.52,0.25	0.57,0.29							
10	0.21,0.11	0.21,0.11	0.23,0.10	0.18,0.14	0.43,0.18	0.32,0.14	0.36,0.21	0.33,0.19	0.36,0.21	0.38,0.19	0.53,0.30						
11	0.21,0.08	0.20,0.06	0.24,0.02	0.19,0.10	0.41,0.05	0.26,0.09	0.37,0.11	0.33,0.10	0.35,0.12	0.39,0.11	0.31,0.09	0.64,0.36					
12	0.27,0.13	0.29,0.10	0.32,0.01	0.24,0.16	0.57,0.06	0.31,0.12	0.50,0.19	0.45,0.18	0.48,0.20	0.49,0.16	0.37,0.16	0.41,0.03	0.76,0.24				
13	0.16,0.09	0.18,0.07	0.20,0.03	0.17,0.11	0.41,0.06	0.21,0.11	0.32,0.14	0.30,0.12	0.31,0.14	0.32,0.11	0.27,0.10	0.27,0.04	0.36,0.06	0.48,0.31			
14	0.24,0.11	0.25,0.09	0.34,0.00	0.23,0.13	0.55,0.02	0.33,0.11	0.46,0.17	0.41,0.15	0.46,0.17	0.46,0.14	0.41,0.15	0.33,0.03	0.43,0.03	0.29,0.04	1.00,0.00		
15	0.15,0.09	0.13,0.10	0.13,0.11	0.11,0.10	0.21,0.19	0.19,0.12	0.16,0.18	0.14,0.17	0.16,0.18	0.19,0.17	0.23,0.13	0.19,0.09	0.21,0.16	0.13,0.10	0.23,0.14	0.35,0.33	
16	0.14,0.05	0.10,0.04	0.17,0.04	0.07,0.05	0.12,0.03	0.10,0.08	0.10,0.04	0.08,0.05	0.12,0.06	0.13,0.04	0.09,0.03	0.13,0.03	0.11,0.03	0.05,0.02	0.18,0.04	0.11,0.04	0.60,0.22

Table 11: Class pair similarity values for sense based similarity

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0.20,0.30																
1	0.08,0.04	0.36,0.38															
2	0.07,0.04	0.13,0.03	1.00,0.00														
3	0.04,0.03	0.03,0.02	0.02,0.02	0.09,0.19													
4	0.08,0.03	0.05,0.03	0.07,0.00	0.05,0.03	0.59,0.41												
5	0.04,0.02	0.03,0.02	0.02,0.01	0.05,0.04	0.07,0.03	0.38,0.44											
6	0.05,0.03	0.04,0.02	0.07,0.04	0.04,0.03	0.13,0.04	0.07,0.04	0.28,0.23										
7	0.04,0.03	0.03,0.03	0.04,0.02	0.04,0.03	0.12,0.04	0.05,0.03	0.18,0.07	0.27,0.25									
8	0.05,0.03	0.03,0.02	0.05,0.03	0.04,0.03	0.13,0.03	0.08,0.04	0.17,0.06	0.16,0.07	0.31,0.32								
9	0.06,0.03	0.03,0.02	0.05,0.01	0.04,0.03	0.12,0.03	0.08,0.05	0.15,0.04	0.12,0.05	0.13,0.05	0.27,0.33							
10	0.06,0.05	0.02,0.01	0.02,0.01	0.04,0.03	0.10,0.03	0.12,0.06	0.08,0.03	0.07,0.04	0.10,0.05	0.10,0.03	0.37,0.36						
11	0.06,0.04	0.03,0.01	0.04,0.00	0.05,0.04	0.08,0.03	0.07,0.06	0.08,0.05	0.06,0.04	0.07,0.05	0.09,0.05	0.07,0.03	0.53,0.47					
12	0.08,0.03	0.04,0.02	0.05,0.02	0.06,0.03	0.10,0.02	0.06,0.03	0.11,0.04	0.08,0.04	0.10,0.03	0.09,0.02	0.08,0.01	0.09,0.02	0.55,0.45				
13	0.03,0.03	0.02,0.01	0.02,0.01	0.04,0.02	0.06,0.02	0.04,0.02	0.04,0.02	0.03,0.02	0.03,0.02	0.04,0.02	0.05,0.03	0.06,0.02	0.08,0.03	0.29,0.41			
14	0.07,0.03	0.02,0.01	0.02,0.00	0.04,0.03	0.06,0.00	0.09,0.04	0.05,0.02	0.04,0.02	0.06,0.04	0.07,0.03	0.14,0.03	0.05,0.01	0.06,0.00	0.03,0.01	1.00,0.00		
15	0.07,0.05	0.04,0.03	0.04,0.04	0.03,0.03	0.10,0.02	0.08,0.06	0.09,0.05	0.08,0.06	0.11,0.06	0.09,0.03	0.12,0.05	0.06,0.03	0.08,0.02	0.03,0.02	0.12,0.05	0.30,0.35	
16	0.06,0.03	0.04,0.02	0.10,0.02	0.03,0.02	0.08,0.02	0.05,0.04	0.08,0.03	0.07,0.03	0.11,0.06	0.07,0.02	0.04,0.02	0.04,0.01	0.05,0.02	0.02,0.01	0.05,0.02	0.05,0.02	0.50,0.26

Table 12: Class pair similarity values for cosine based similarity

The former is for sense similarity while the latter is for cosine based similarity. The mean and variance of similarity for two given classes C_i and C_j, of documents are given by,

$$Mean(i,j) = \frac{\sum_{d_i,d_j \in C_i,C_j} sim(d_i,d_j)}{|C_i| \times |C_j|}$$

$$Variance(i,j) = \sqrt{\frac{\sum_{d_i,d_j \in C_i,C_j} (Mean(i,j) - sim(d_i,d_j))^2}{|C_i| \times |C_j|}}$$

The tables 11 and 12 are represented by plotting the contour graphs and wireframe models. For each pair of classes we plot the contour graphs of the mean similarity values computed using both the cosine and the sense based similarity measures. In these graphs, the X and Y axes represent the classes and the colours represent the similarity gradients. Figure 7.2 represents the contour map for sense based similarity and figure 7.2 represents that for the cosine similarity values. Similarly, the figures 7.2 and 7.2 represent the wireframe graphs for the mean of classes based on sense based similarity and cosine based similarity respectively.

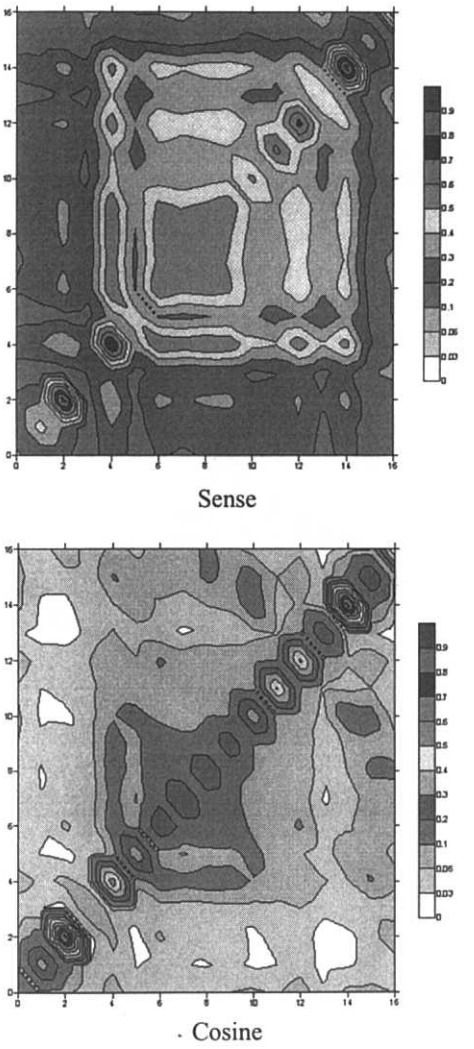
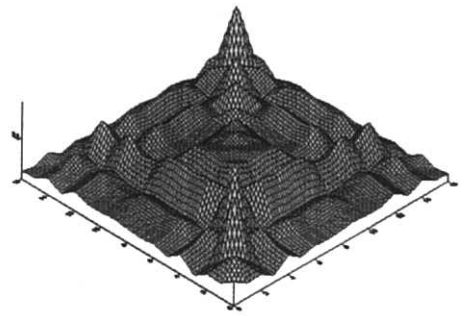
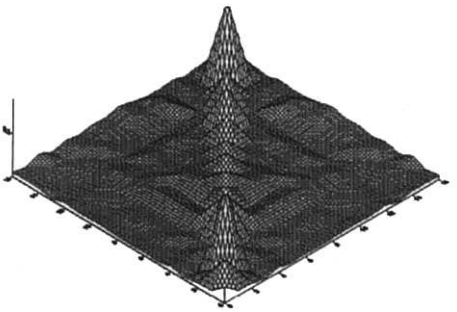


Figure 6: The contour plot for cosine vs sense based similarity



Sense



Cosine

Figure 7: The wireframe plot for cosine vs sense based similarity

Discussion of Results for Setting-1 Experiments

Let us try to examine the properties of the matrices that we have obtained in tables 11 and 12. These lower triangular matrices represent the inter-class mean similarity values. For every tuple represented in a cell of the table, the first part stands for the mean and the second part for the variance.

It is seen from the diagonal of the matrix that the mean similarity values exceed the values in the same column and in the same row. This is expected since the diagonal values are self similarity numbers. We also note that the mean self similarity values is high in class 4 which is *Science Fiction* as well as in class 12 which is *Management*. This is due to consistent use of almost the same set of nouns across documents within the class. The mean self similarity is pretty low in class 3 and class 15 which are *Science* and *Religion* respectively. The former is due to the fact that *Science*, as a class is extremely diverse with multiple disciplines in it which have their specific terminologies. In the case of *Religion*, nouns typically are used in very many senses. In all the cells we observe that the variance is small ranging from 0.05 to 0.3, which means that the similarity values are not widely dispersed. This gives us confidence in the mean similarity value as a measure.

Now, we make some interesting observations about the cross similarity values. Class 6 is an interesting point. The similarity among classes 6, 7, 8 and 9 are close to their self similarity values. These documents all belong to *Fiction* category- *Mystery*, *Adventure*, *Romance*, and *Humour* respectively. Works of fiction tend to use similar words and phrases. This fact leads to the high cross-similarity values. This observation is further vindicated by the behaviour of the classes 0, 1 and 2 which are *Politics*, *Law* and *Taxation* respectively. Documents in these areas tend to use similar terminology. Coming to low cross-similarity values we see this happening in the row 16 of table 11. Documents in class 16 pertain to *Sports* and hence share few terms with documents of other classes like *Religion*, *Literature*, *Science* etc.

The above observations based on numbers are brought home very effectively with the contour maps and the wireframe diagrams shown in figures 6 and 7. The region around the center of the contour map spanning coordinates 5-9 in both X and Y axes shows reasonably high similarity value in brown colour. These are the documents from literary classes as mentioned above. When one steps out of the region bounded between 3 and 15 on either axes, one observes a sharp decrease in the similarity values. This transition is into sports documents on one side, and politico-legal documents on the other side. The difference in literary styles and terminology choices are brought out by this factor.

Comparison

One immediate observation from the tables, contour maps and wireframes is that the similarity values are shown in a more enhanced and sharper fashion by the sense based similarity measure. We see more details in the contour map and the wireframe for sense based similarity, than in those for cosine similarity. For example, the terrain between $X = 7$ and $X = 11$ show more separate regions in the wireframe diagram for sense based similarity. This highlights difference among the similarity measures between classes like Psychology, Romance and Humour. On the other hand, if we follow the diagonal we see better separated regions in the cosine similarity measure based diagrams. This measure seems to better highlight the difference between self similarity values.

On comparing the two graphs we find that the similarity values are more uniformly dis-

tributed over the documents in the case of sense based similarity measure. Also, we find that distinct square regions of high similarity are formed among the documents of the same class. This is not so prominent in the case of cosine similarity.

7.3 *Setting 2: K-NN classification*

In setting-1, we just looked at the qualitative aspect of the similarity measures obtained and compared our similarity measure with that of cosine similarity. In this section we try to obtain a quantitative evaluation for our similarity measure.

From the documents given in table 10, the nouns were extracted. For one experiment we chose the senses of these nouns from the Semcor itself whereas in the other experiment we passed these nouns to the disambiguator module. These correspond to the situations of *fully correct sense* and *partially correct sense* respectively. The disambiguated nouns in each case were passed finally to the similarity estimator. From these we got two sets of inter-document similarity values.

A similar procedure was carried out to get the inter-document cosine similarity values. In order to meaure the goodness of these inter-document similarity values, we used these values to classify the documents using the K-Nearest Neighbourhood (K-NN) classification method. In this method some prototypes are fixed as representing a class. When a test case is encountered it is compared with the prototypes and K nearest prototypes are selected. The class of the test case is decided by the class to which the majority of its neighbours belong. Here, out of the 99 documents in the corpus, 75% were used as prototypes of classes and the rest as test cases. The result of these trials are given in figure 8.

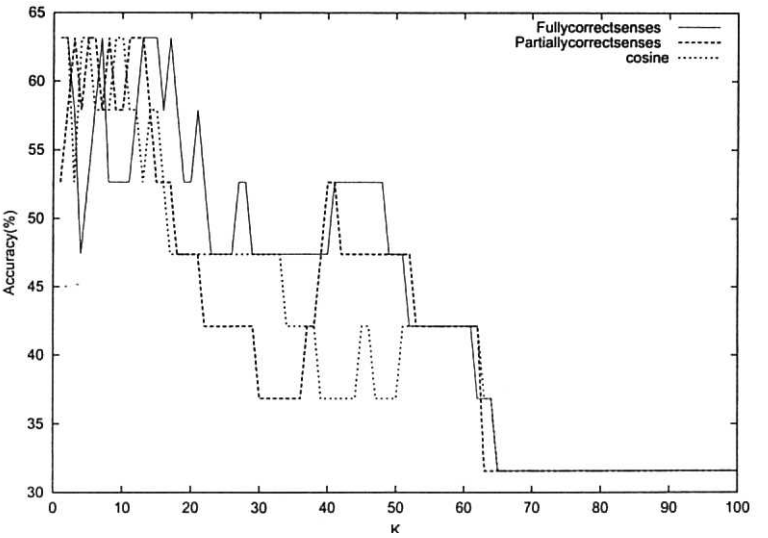


Figure 8: Plots of the K-NN classification

The three cases that we had discussed have been shown in the graphs. The case where we have taken the senses directly from the tags in Semcor corresponds to the one marked as *Fully correct senses*. This is so, because the word senses provided are manually tagged ones

and are expected to be 100% correct. The one marked *Partially correct senses* corresponds to the trial where we have disambiguated senses of the nouns using our algorithm for word sense disambiguation. The third one marked *cosine* corresponds to the cosine similarity.

Discussion of Results for Setting-2 Experiments

As we can see, the trend is not very clear from the graphs. For small values of K the accuracy is quite high, and both the measures perform equally well. For values of K between 20 and 50, the sense based similarity is better than the cosine similarity for the first case of fully correct senses. But after K exceeds 50, all the three plots plummet to a constant accuracy. This of course is due to the fact that the value of K far exceeds the cardinality of the training set. When K exceeds the cardinality of training set, all the documents in the training set are considered neighbours of the test document and there is no change with the increase in K beyond that point.

7.4 Setting-3: Clustering

Unable to draw any firm conclusions due to the uncertainty in the value to be chosen for K, we preformed clustering experiments. We employed *Hierarchical Agglomerative Clustering*[5]. It is a method of clustering in which clusters are built hierarchically by merging existing clusters with the highest similarity to form new clusters. The basic requirement for this clustering is that the similarities between the documents be available. The steps of the clustering technique are:

- Initialize clusters with the individual documents, *i.e*, the initial cluster consists of just a single document.
- Choose the cluster pair with the highest similarity and merge them into a single cluster.
- Update the similarity values between the newly formed cluster and the rest of the clusters.
- Repeat this process till the required number of clusters are formed.

The third step is the most important where we define inter-cluster similarity using the similarity values of the documents constituting the cluster. Various measures can be used for this. The most commonly used is the maximum similarity value between the constituents of the two clusters. Expressed mathematically it is,

$$Sim(A, B) = max(Sim(A \times B))$$

where A and B are two clusters and $A \times B$ is the cross-product of the two clusters.

This algorithm was used to cluster the corpus of 99 documents derived from Semcor under the three test criteria discussed earlier.

Though the clustering process is basically an unsupervised approach, to measure the accuracy of clustering, we need to label the clusters and find the number of documents that have been wrongly classified. In the above algorithm this was done by stopping the hierarchical building process when there were 17 (the number of classes in the corpus) clusters. The clusters were labeled by the majority class of the documents in the cluster. The rest of the documents were then assigned the class corresponding to the cluster to which it belongs. This was then compared with the actual classes of the document. The accuracy of clustering is then based on the number of matches.

Algorithm	POS accuracy(%)	WSD accuracy(%)	Search_depth	MAX	Clustering accuracy
Fully correct sense	100	100	-	3	37.37
Fully correct sense	100	100	-	2	36.36
Partially correct sense	100	47.01	3	3	33.33
Partially correct sense	100	47.01	3	2	39.39
Cosine	-	-	-	-	35.35

Table 13: The accuracy of clustering

Discussion of Results for Setting-3 Experiments

From the above experiment we find that the clustering accuracy is the highest for the case where we have the correct word senses. The first entry in the table 13 represents the ideal case where we have cent percent accuracy for all the modules. Also, we see that by varying the parameter **search_depth** we can increase the accuracy of the word sense disambiguation and hence increase the accuracy of the similarity algorithm. The role of the parameter **MAX** in relation to the accuracy of the similarity algorithm is not very clear, but a few words are in place regarding the two parameters in the system and their influence on the working of the algorithm.

7.5 Influence of Search_Depth and Similarity_Radius

The two parameters *search_depth* and *MAX* stand for the same property and are related to the control of similarity in the WordNet graph. By specifying a maximum radius of influence of the similarity measure, we are able to control the definition of similarity. The more the value of *MAX* or *search_depth*, the more is the dilution of similarity, leading to a reduction in precision. This is very evident in the behaviour of *search_depth* in determining the precision-recall balance in word sense disambiguation. In the case of *MAX*, it influences the noisy links obtained during the similarity measurement. When we go on increasing this parameter there is a high probability that a path is found between two words. This brings in a lot of noise in the form of useless links. Although the links with higher path length are penalized, the sheer number of such links overwhelms the measure. On the other hand if we minimize the value of *MAX*, in the extreme case of $MAX = 0$ it reduces to the cosine similarity. It then becomes important for us to determine the optimum value for these parameters. Typically it can be assigned a value less than the average radius of the WordNet graph.

8 Conclusion and future work

From our evaluation report of the three part experimentation it is clear that the proposed similarity measure based on the *resources of the WordNet* is highly promising. It highlights the similarity where it should, leads to better K-NN classification for appropriate values of K and achieves superior clustering- all in comparison to the classical cosine similarity measure. Thus this work should be looked upon as an attempt to **introduce semantics into similarity determination**. By doing this we have tried to bridge the gap between the linguistics based text understanding and the task of information retrieval. Information retrieval has traditionally

been dominated by stochastic methods for similarity measurement. Through the implementation of the system we have demonstrated the use of lexical knowledge and semantics in information retrieval. It must be stressed that the work done here just forms the preliminary stage to the integration of meaning conveyed by a text into the information retrieval applications. There is still a long way to go before we can actually harness the full information conveyed by the text. Recent developments like **UNL**[15] promise to deliver the results which we require to solve these problems.

The future work consists in:

- Improving the efficiency of the similarity computing algorithm through mechanisms of faster access to the WordNet data structures.
- Increasing the algorithm performance by involving words of parts-of-speech other than noun also. Here a verb centric sentence representation scheme like the Universal Networking Language[15] will be of great help.
- Enhancing the accuracy of the word sense disambiguation module. Any improvement in the WSD accuracy directly impacts the accuracy of the similarity measure.

References

[1] Eneko Agirre and German Rigau. A proposal for word sense disambiguation using conceptual distance. In *Proceedings of the First International Conference on Recent Advances in NLP*, 1995.

[2] Christiane Fellbaum, editor. *WordNet, An Electronic Lexical Database*. The MIT press, 1999.

[3] Francis and Kucera. *Computatinal Analysis of present day American English*. Brown University Press, 1967.

[4] S. Green. Building hypertext links by computing semantic similarity. *IEEE Transactions on Knowledge and Data Engineering*, 11(5):713–730, 1999.

[5] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs NJ, U.S.A., 1988.

[6] J. Lee, M. Kim, and Y. Lee. Information retrieval based on conceptual distance in isa hierarchies. *Journal of Documentation*, 49:188–207, 1993.

[7] Oliver Mason. Qtag: A portable parts of speech tagger. http://www.clg.bham.ac.uk/QTAG/, 1998.

[8] G. Miller, C. Leacock, T. Randee, and R. Bunker. A semantic concordance. In *proceedings of the 3rd DARPA Workshop on Human Language Technology*, pages 303–308, 1993.

[9] J. Morris and G. Hirst. Lexical cohesion, the thesaurus, and the structure of text. *Computational Linguistics*, 17(1):211–232, 1991.

[10] R. Rada, H. Milli, E. Bicknell, and M. Blettner. Development and application of a metric on semantic nets. *IEEE Transactions on Systems, Man and Cybernetics*, 1(9):17–30, 1989.

[11] Philip Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research (JAIR)*, 11:95–130, 1999.

[12] Gerald Salton, editor. *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer*. Addison-Wesley, 1989.

[13] Michael Sussna. Word sense disambiguation for free-text indexing using a massive semantic network. In *Proceedings of the Second International Conference on Information and Knowledge Management*, 1993.

[14] D. Yarowsky. Word sense disambiguation using statistical model of roger's categories trained on large corpora. In *Proceedings of the 15th International Conference on Computational Linguistics*, 1992.

[15] Meying Zhu and Hiroshi Uchida. Universal networking language specification. Technical report, UNU/IAS/UNL Center, 1998.

Real World Semantic Web Applications
V. Kashyap and L. Shklar (Eds.)
IOS Press, 2002

Semantic Enhancement Engine: A Modular Document Enhancement Platform for Semantic Applications over Heterogeneous Content

Brian Hammond, Amit Sheth*, Krzysztof Kochut*
Semagix, Inc.
Also, LSDIS Lab, Computer Science, University of Georgia

Abstract. Traditionally, automatic classification and metadata extraction have been performed in isolation, usually on unformatted text. SCORE Enhancement Engine (SEE) is a component of a Semantic Web technology called the Semantic Content Organization and Retrieval Engine (SCORE). SEE takes the next natural steps by supporting heterogeneous content (not only unformatted text), as well as following up automatic classification with extraction of contextually relevant, domain-specific (i.e., semantic) metadata. Extraction of semantic metadata not only includes identification of relevant entities but also relationships within the context of relevant ontology.

This paper describes SEE's architecture, which provides a common API for heterogeneous document processing, with discrete, reusable and highly configurable modular components. This results in exceptional flexibility, extensibility and performance. Referred to as SEE modules (SEEMs), which are divided along functional lines, these processors perform one of the following roles: restriction (determine the segments of the input text to operate upon); enhancement (discover textual features of semantic interest); filtering (augment, remove or supplement the features recognized); or outputting (generate reports, annotate the original, update databases, or other actions).

Each SEEM manages its configuration options and is arranged serially in virtual pipelines to perform designated semantic tasks. These configurations can be saved and reloaded on a per-document basis. This allows a single SEE installation to act logically as any number of Semantic Applications, and to compose these Semantic Applications as needed to perform even more complex semantic tasks.

SEE leverages SCORE's unique approach of creating and using large knowledge base in semantic processing. It enables SCORE to provide flexible handling of highly heterogeneous content (including raw text, HTML, XML and documents of various formats); reliable automatic classification of documents; accurate extraction of semantic, domain-specific metadata; and extensive management of the enhancement processes including various reporting and semantic annotation mechanisms. This results in SCORE's advanced capability in heterogeneous content integration at a higher semantic level, rather than syntactical and structural level approaches based on XML and RDF, by supporting and exploiting domain specific ontologies. This work also presents an approach to automatic semantic annotation, a key scalability challenge faced in realizing the Semantic Web.

1. Introduction

Today, many organizations have access to a broad variety of information resources, including collections of internal documents, data repositories, and external public and subscription-based content. Collectively referred to as enterprise content, these information resources occur in a wide variety of formats and digital media. In order to truly take

advantage of enterprise content, the next generation of enterprise content management systems require novel, advanced capabilities.

Making information resources searchable is a step in the right direction, but to get the most out of these resources, users must be able to access information through mechanisms that blend seamlessly into the workflow and applications they use. The value of internal resources may be significantly enhanced when coupled with open source information on the World Wide Web, private networks, and the deep web.

Additionally, it would be highly valuable to establish links between semantically related internal and external documents. For this, documents must be converted from simple text buckets to collections of meaningful information that can be collated with web resources. In other words, traditional document management needs to be enhanced with knowledge management and semantic technology.

By combining classification with syntactic and semantic metadata extraction, the SCORE (Semantic Content Organization and Retrieval Engine) Enhancement Engine (SEE) component has been designed to do just that. The process of automatic classification puts the user in the ballpark, and contextual metadata extraction puts him in the seat; or, in the parlance of the World Wide Web, classification puts the user in the web ring, and metadata puts him on the web page.

Systems that can be built around this type of functionality are as diverse as the documents with which they work. SEE is a system designed with flexibility and integration issues as high priorities from the ground up. SEE's implementation centers around three primary concepts: core document service, modular processing, and superior configurability.

Core document processing tasks, such as format normalization (e.g., tag versus text) and linguistic analysis (e.g., sentence boundaries), are needed to address the issue of format variety. Modular processing and configurability address the wide variety of workflows into which SEE must be integrated, as well as providing a mechanism through which classification and metadata extraction may be coupled.

This allows for the extraction of domain-specific metadata from raw text (Figure 1). First, the classification technology determines the category for a document and hence determines the domain of discourse. Then semantic metadata particular to the domain is targeted and extracted. This includes specific named entity types of interest in the category (such as "CEO" in "Business," "Chipset" in "Technology," or "SideEffects" in "Pharmacology") as well as category specific, regular expression-based knowledge extraction. This domain-specific metadata can be regarded as semantic metadata, or metadata within context. It is possible to associate multiple domains with a document and extract different semantic (domain-specific) metadata for that document. It is also possible to take more than one pass of the process shown in the figure. The classification and extracted metadata forms the basis of semantic annotation of the documents. Although currently an XML based format is used for annotation, in future RDF and OWL can be easily supported.

The automatic extraction of semantic metadata from documents which have not been previously associated with a domain is a unique feature of SEE. In essence, this transports the document from the realm of text and mere syntax to a world of knowledge and semantics in a form that can be used for computation.

By providing a framework for working with the relevant knowledge in a document rather than the text, SEE allows for a new breed of semantic applications to be implemented in one easy-to-use, extensible package. HTML documents can be moved from the World Wide Web to the Semantic Web [1] through the process of semantic annotation that attaches automatically extracted semantic metadata to text.

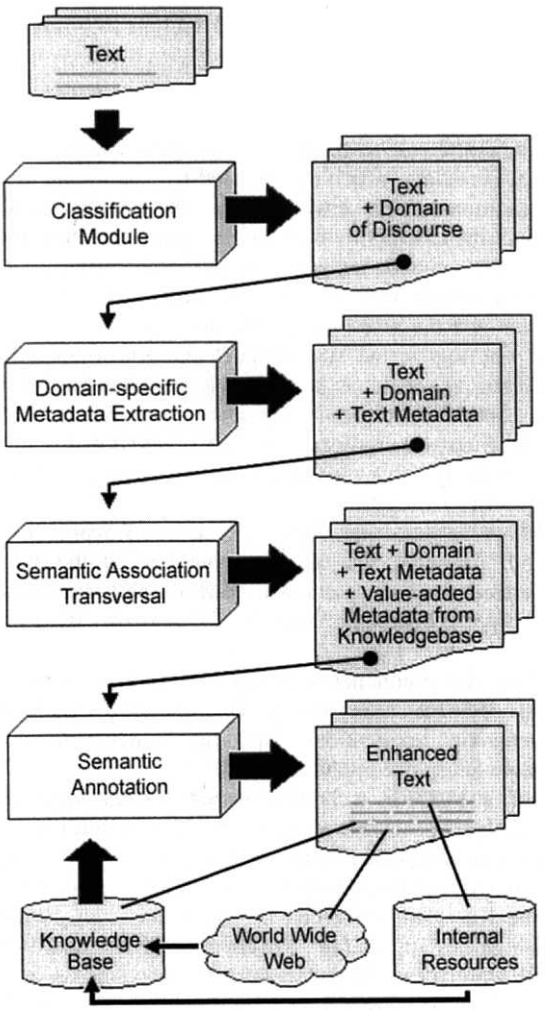


Figure 1: Through the use of automatic classification and contextually relevant KnowledgeBase information, domain specific metadata can be extracted from a document, enhancing the meaning of the original and allowing it to be linked with contextually heterogeneous content from multiple sources.

To summarize, the unique features of SEE are: core document services with a modular functionality; the use of automatic classification to allow for domain-specific metadata (semantic metadata) extraction; and a simple, unified format for configuration of modules, which allows a single installation of SEE to be used for an immense variety of applications.

This chapter is organized as follows. Section 2 provides a relevant overview of the SCORE technology, of which SEE is a component. Section 3 provides an overview of SEE. Section 4 discusses core document services SEE supports. Section 5 discusses four types of SEE modules (SEEMs) that support respective semantic tasks. Section 6 discusses configuration and implementation features. Section 7 discusses Virtual Machines, which support basic Semantic Applications, such as domain specific metadata extraction and integration of internal and external knowledge sources, by combining various SEEMs. Section 8 provides additional details on several SEEMs and an example of automatic

semantic annotation, which includes identification and extraction of contextually relevant relationships. Section 9 discusses related work, and Section 10 offers conclusions.

2. Background: SCORE Technology

Ontologies play a central role in most semantic technologies. They form the basis for syntactic and semantic metadata, which can be used for annotating or tagging content. The content's context determines which semantic metadata to extract. Automatic classification technology helps select the context by classifying documents into one or more categories and extracting or inferring semantic metadata corresponding to one or more contexts.

We have divided ontology into two related components: the definition component, called the WorldModel and the assertional component called the Knowledgebase. As with the specification for ontologies, the WorldModel and Knowledgebase definition process involves domain-specific expertise as well as an understanding of eventual application requirements. While some clustering techniques can provide initial input to semi-automate this process [2, 3], it cannot generally be completely automated if high-quality results are needed. Also the techniques that rely on learning from structured data [3] are not particularly practical.

The Knowledgebase reflects the subset of the real world for which a semantic application is created. As such, it is an important part of the solution. It enables the extraction of value-added semantic metadata, such as the ticker symbol "INTC" when a document only mentions the company "Intel." It also provides the framework for semantic associations.

In addition to these two components, SCORE provides a query-processing system. A comprehensive suite of APIs uses this query-processing capability to support rapid development of semantic applications such as search, directory, personalization, syndication, and custom enterprise applications [4].

The operation of a SCORE technology-based system involves three independent activities, as illustrated by the sections separated by the gray lines in Figure 2 [5]. Defining the WorldModel and Knowledgebase is the first activity (Figure 2, top right). Knowledge extraction agents manage the Knowledgebase by exploiting trusted knowledge sources. Different parts of the Knowledgebase can be populated from different sources. Various tools help detect ambiguities and identify synonyms. Commercial deployments of SCORE can be expedited with a predefined WorldModel and Knowledgebase.

Content processing comes second (Figure 2, left). This includes classifying and extracting metadata from content. The results are organized according to the WorldModel definition and stored in the Metabase. Knowledge and content sources can be heterogeneous (XML, RDF, static and deep Web pages, database, or documents in various formats), internal or external to the enterprise, and accessible in Push (content feeds or database exports) or Pull (Web sites or database queries) modes.

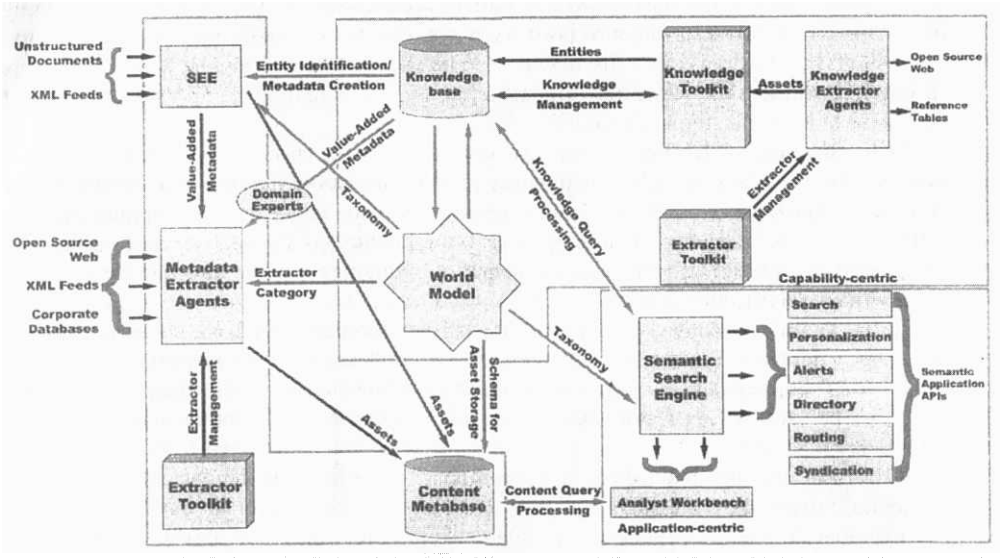


Figure 2: SCORE system architecture. The three activities bounded by gray lines cooperate through XML-based knowledge and metadata sharing.

Support for semantic application comes last (Figure 2, bottom-right and top left). The Semantic Search Engine (SSE) processes semantic queries and returns results in XML from the Metabase. An API for building traditional and customized applications facilitates GUI creation. SEE provides a modular framework for domain-specific metadata extraction through the use of automatic classification techniques.

3. SEE Overview

SEE provides a variety of services and functions as well as a framework for implementing semantic applications.

A semantic application may involve a number of inter-related *semantic tasks*. Examples of semantic tasks include text classification, metadata extraction, identification of semantic associations, and enrichment of a document by annotating it with pertinent semantic information. The SEE framework allows for an easy integration of such semantic tasks in order to achieve a more complex objective. As an example, such semantic tasks may be organized into a pipeline to create a large semantic application in which text documents obtained from a network live feed are first classified into a node in taxonomy. Subsequently, entities relevant to the determined category of the document (in the Knowledgebase) are identified. Furthermore, relationships existing among them are uncovered, and other entities that may be important, though not explicitly present in the document, are identified via known relationships. Ultimately, the original document is enriched by suitable annotations, including the category of the document, recognized entities, identified relationships and other relevant entities. A detailed description of some of the semantic tasks is presented in Section 8.

The core concepts behind building systems based on SEE include:

Core Document Services: A modern document system must be able to handle XML, HTML, raw text and various other document formats[1]. The text segments must be

[1] We assume use of format converters such as Verity's Keyview [http://www.verity.com/keyview/] and Stellent's Inso/Outside In [http://www.stellent.com] to help obtain text from proprietary document formats.

decomposed into chunks, such as words, phrases and sentences. Access methods for going from character to word to sentence position are provided to maintain maximum flexibility. Text annotation methods allow for new data to be added easily to existing documents. This allows SEE module implementers to concentrate on implementing text algorithms without the hassle of basic parsing and character offsets.

SEE Modules (SEEMs): Each text-processing task performed in the system is implemented as a SEEM. The functionality provided usually falls into one of four major divisions: restriction (determine the segments of the input text to operate upon); enhancement (discover textual features of semantic interest); filtering (augment, remove or supplement the features recognized); or outputting (generate reports, annotate the original, update databases, or other actions).

These modules are then arranged into pipelines through which incoming documents flow. As a document passes through the modules, text segments of semantic interest are identified. These segments of text are referred to as DocumentFeatures. DocumentFeatures may also be attached to a document based on features found or other criteria such as automatic classification.

When Entities are discovered in a document, other Entities can be attached to the document through the traversal of semantic associations. These traversals can be single-step (for example "Linux" has "created by" relationship with "Linus Torvalds") or may follow user-defined paths (such as "Microsoft" has a "competes with" relationship with "Sun Microsystems" which has a "created" relationship with "Java.") This allows for connections to be created between entries in the Metabase by exploiting domain-specific knowledge. This represents a unique functionality of SEE and SCORE.

Modular Configurations: All SEE modules use a common configuration API that produces XML output, which provides information about the use, type and acceptable values for its options. By using XSLT, different aspects of this configurability can be exposed via a GUI to the end user. These configurations can also be saved on the SEE server to be later loaded by name on a per-document basis. In addition to the module options, their order in the pipeline is also stored. Since SEE is HTTP based, configuration commands can be added to hyperlinks to facilitate custom annotation.

Virtual Engines: Each configuration of SEE can be thought of as a discrete functional unit. This allows a single installation to be used by scripts as though it were in fact several distinct semantic applications. This facilitates the creation of systems that perform advanced semantic processing, such as the extraction of category metadata extraction based on automatic classification, the linking of HTML documents to various internal and external resource by metadata type and other applications which can benefit from performing additional Semantic processing based on semantic data. The effective use of semantic feedback is a unique and powerful property of SEE.

More detailed information about core enhancement modules is provided in the remaining sections to demonstrate the functionality and usage of the system. An example of the HTML annotation with semantic metadata is also given.

4. Core Document Services

The importance of the World Wide Web as a content source makes it imperative that the system be able to handle documents that are in HTML. The emergence of the Semantic Web mandates that SEE also be able to support XML (and in the future, RDF) as well. In SEE, before any module processes a document, it must first pass through three parsing phases: format parsing, word parsing and sentence/phrase parsing.

Format parsing is primarily concerned with the separation of tags from text. Despite their similarity and common ancestry, HTML is much more problematic to work with than

its younger cousin, XML. Though at first glance this may sound like a non-issue because it is an operation so commonly performed, the parsing of HTML represents a number of challenges.

Ironically, the problem seems to have arisen from the aspirations of HTML parser writers in the past to make life easier for those who deal with HTML. In the beginning HTML browsers, such as Mosaic, were fairly strict and would complain about small mistakes in HTML authorship. At that point in time, most HTML documents were still written by humans directly rather than through authoring tools, so simple errors were a common source of annoyance for web surfers.

As browsers improved their ability to deal with partially malformed HTML, authors have become less and less concerned with fixing mistakes, or were perhaps simply unaware of their existence. All of these factors have led to very nice HTML parsing in browsers but also to many rather nasty quirks for the implementers of new parsers. By contrast, XML is much easier to work with because it imposes tighter constraints on well-formedness.

During the next two parsing phases, SGML entities representing ampersand, quote and other punctuation symbols must be resolved and noted. They must be resolved so that their true meaning as punctuation can be maintained, and they must be noted so that character offsets can be adjusted accordingly.

This can be best illustrated by a brief example: "AT&T is a telecommunications company." The text logically references "AT&T", which is four characters in ASCII, but is written using eight. This becomes an issue when it comes time to identify DocumentFeatures and later when it is time to perform semantic annotation.

Once those portions of the document that are markups are identified, the next step in the document preparation phase is to segment the text into individual words. Words are always broken up by white space and often by punctuation. However, there are cases in which punctuation is considered integral to the word itself. Some examples include: O'Connor, AT&T, and Matthews-Morgan. This results in a list of word tokens that is then passed into the sentence/phrase analyzer.

When identifying DocumentFeatures, some false positives can be avoided by using end-of-sentence and end-of-phrase information. SEE currently uses a number of heuristics based primarily on punctuation and capitalization to detect breaks corresponding to sentence and phrase boundaries. Though currently designed for English syntax, the SEE sentence/phrase analyzer is being enhanced to support other languages; this is made easier by its object-oriented design. This can be contrasted with systems that use table-driven parsing.

5. SEE Modules

Once a document has been prepared, it is sent to the SEE Pipeline. Each Pipeline is composed of a number of SEEMs. Control flows from SEEM to SEEM with different parts of the semantic tasks performed by different modules. The individual modules perform discrete tasks that usually fall into one of the following roles: *restriction, enhancement, filtering* and *outputting*.

Restriction Modules: The purpose of modules that perform restriction tasks is to identify the portion or portions of a document that should be enhanced. Many web pages, for example, have sidebars with text unrelated to the main story. To address this issue, there is a SEEM which attempts to identify the primary text. It then informs SEE to ignore the rest of the document for enhancement purposes. Similarly, often only certain elements of an XML document should be examined.

In the interest of efficiency, SEEMs that restrict are typically placed in the pipeline **before** the built-in sentence/phrase boundary detection component. This is because there is no point in performing such analysis just to ignore it.

Restriction modules are unique in that they are almost the only SEEM which needs to be concerned with the original document format to fulfill their task.

Enhancement Modules: As might be expected from the name, SEEMs that fall into the enhancement category are the métier of SEE. The primary purpose of such modules is to identify DocumentFeatures that occur in the original text.

Usually, a DocumentFeature is a matched region of text placed within a semantic context. Examples of such DocumentFeatures include the proper names of people, places and things (named entities); monetary sums, percentages and dates (unnamed entities); and phrases.

There are other kinds of DocumentFeatures not directly associated with a particular section of the document. Examples of such features include classification results, relationships between named Entities found in the text, and Entities that may be added to the text by contextual rules based on Entities found in the input document. Filter or output modules usually add the latter. These DocumentFeatures represent metadata for the document that are the basis of semantic annotation.

Filtering Modules: The primary purpose of filtering modules is to modify the set of DocumentFeatures generated by enhancement modules. These modules can be thought of as the "decision makers" of the SEE. One of the most important purposes of filtering is the removal of semantic ambiguity. In SEE, this can occur when a single region of text matches multiple instances, such as "John Smith".

By using relationships between Entities found in the text and knowing the domain of the document through classification, SEEMs can make reasonable decisions about which of many potential Entities to keep and which to discard. Filter modules may also add additional DocumentFeatures based on other recognized features, such as adding the name of a company when its CEO is mentioned in a document.

Output Modules: One or more output modules produce the final result of a trip through the SEE pipeline. Output typically takes the form of report generation or annotation of the original document. In the former, a new document is created with information about the DocumentFeatures found in the input document. Reports might be in XML, plain text or SQL, depending on the application.

By contrast, in the case of annotation, the original text is augmented with semantic metadata added in line with the sections of text with which it is associated. In the case of HTML, this may take the form of new hyperlinks to repositories containing detailed information-recognized Entities or, in the case of classification, possibly links into the spot in a directory containing similar documents. The WorldModel or one of its components (for example, a subtree rooted in the "Sports" node of the WorldModel) in SCORE may be used as the domain ontology.

6. Modular Configurations and Other Implementation Issues

Each SEEM is compiled into a separate Dynamically Shared Object (DSO) file. Each time SEE starts, it loads and initializes each SEEM from its corresponding DSO file. If a SEEM fails to initialize, the problem is logged, and initialization of SEE continues. In the case of a faulty or misconfigured module, the rest of the system will still remain available. Compiling a SEEM into a separate executable file results in many advantages: additional

functionality can be added to the existing installation in a straightforward, extensible way; bugs can be fixed simply by swapping out modules; and new functionality can be added by extending a C++ class, which helps reduce development time for extending functionality.

Due to the plug-and-play nature of SEEMs, it is necessary to provide a common mechanism so that new modules can be added to an existing installation without affecting the current SEE or requiring it to be upgraded. To this end, all SEEMs make use of a common configuration API. This API produces XML that describes the purpose of the module as well as information about its options. The description of each option specifies its purpose, value type (string, integer or boolean), range of accepted values, and the default and current value of all of the module variables.

When SEE processes a request to send its current configuration, it returns an XML description that includes a list of saved configurations, the modules available, and the modules currently in the pipeline. Then SEE calls a method for each SEEM which produces its tasks-specific XML description. This allows SEE modules to be largely self-documenting and easily configured. Configuration of SEE is highly customizable by using XSLT. Dynamic GUIs can be created on a per-user basis so that novice users need not be overwhelmed by options they do not want to consider or do not need to change. Additionally, the GUI can be implemented to closely match the look and feel to which users have become accustomed, adding to the comfort level.

Configurations can be saved on the server with SEE for later use. These configuration files can be referenced by name and invoked on a per document basis. Once a configuration is loaded, the state of SEE and its modules remains set until another configuration file is loaded or a configuration command sent. Because SEE saves a copy of the last valid configuration loaded to disk and re-reads it on initialization, it retains its state even in the event of a catastrophic system failure, such as a power outage.

The use of named configuration files also allows a simplified interface consisting of a drop-down menu box, where users can select canned configurations for different purposes. In this way, SEE can support a wide range of applications as middleware, where the end user does not even need to be aware of the full range of options available to be configured. More importantly however, since the configurations can be invoked on a per document basis, and because SEEMs can implement such a diverse range of semantic functionality, this provides a mechanism whereby a single SEE can be used for completely different purposes. Logically, each configuration can be thought of as a different application instance running under SEE.

7. Virtual Engines

By thinking of each configuration of SEE as a distinct functional unit, the utility of SEE is dramatically expanded. Not only can SEE perform a wide variety of semantic tasks, but through the use of scripts, these SEE application instances can be chained together to create pipelines for semantic processing. Since each step in the pipeline is a semantic application, the meaning of the metadata can be maintained throughout the entire process. This allows the creation of systems that can make decisions and take action within the context of meaning rather than data. This allows for semantic application beyond annotation, where semantics can play a role in the decision making process for automated agents.

When a document comes into SEE, it is first identified as a collection of words and sentences. Then, semantic metadata is recognized and added in the form of DocumentFeatures. At this point, the document has been augmented with knowledge. To make the most of this augmentation, applications must have the semantic savvy to exploit it. To illustrate this concept, consider the following scenarios.

Domain Specific Metadata Extraction: The *RightNow News Service* has a stream of documents coming in from reporters around the globe. The documents are short, two or three paragraphs of text each, and fall into one of twenty possible categories. The goal is to classify and extract category specific metadata for each incoming document.

Metadata to be extracted is based on the relevant domain specific ontology. For documents that fall into the financial category, the system needs to extract dates, monetary amounts, companies, CEOs and ticker symbols. If a document is classified as baseball, the names of teams, players, managers and scores should be identified. For other categories, domain specific metadata can also be discovered.

Because SEE integrates classification and metadata extraction, an application like this can be implemented very quickly. Quite often, companies and groups interested in classification and per category metadata will have amassed large collections of documents that have been classified already. These corpora can be used to train the automatic classification subsystem of SEE.

Integrating Internal and External Knowledge Resources: Despite the fact that *TransMeta Investment World* (TMIW) analysts have access to an extensive collection of in-house financial data, many of them still find external web resources extremely useful. Always on the lookout for competitive advantage, TMIW wants a way to combine these two sets of resources.

HTML documents can be annotated with links to a local Knowledgebase for Companies, a real-time streamer for stock symbols and hyperlinks to *biography.com* for company executives. SEE can accomplish this contextual annotation by processing a given web page multiple times with slightly different configurations controlling the type of Entities recognized each time.

Because SEE can switch gears on a per document basis, a single engine could be used to perform both of these completely different tasks. Since they are based on HTTP, SEE configuration commands can be "embedded" in hyperlinks, even in hyperlinks generated by SEE HTML annotation.

8. Selected SEE Modules in Detail with Examples

Since the primary purpose of SEE is to link documents with knowledge, the SEEMs that perform enhancement are perhaps the most interesting. A module known as the eMDExEnhancer recognizes named Entities. Dates, percentages and monetary amounts are matched by the eRegexpEnhancer. Potentially interesting regions of text are located by the ePhraseEnhancer for later analysis. Automatic classification is done by the eClassifyEnhancer.

The eMDExEnhancer

The Knowledgebase in SCORE keeps track of named Entities such as people and places [5]. The eMDExEnhancer is the module responsible for matching regions of text with entries in the Knowledgebase.

This is done using a number of pattern matching rules specific to various types of Entities. Person names, for example, are usually stored in Knowledgebase in the canonical form of: "LAST_NAME, FIRST_NAME MIDDLE_NAME*(, SUFFIX*)?".

In order to accomplish name recognition, this form must be mapped to a more natural language flavored version. Aliases, also called nicknames, must be considered as well.

The eMDExEnhancer makes a number of passes over each document. In the first pass, only the most probable matches are retained. So in a case like: "John Black is a man. Black wears pants," the first pass would identify "John Black" in the first sentence but not "Black" in the second. Once this stage is completed, probable references such as "Black" are then associated with their most likely Entities from stage one.

Obviously, this process if fraught with pitfalls, and a number of measures have been taken to avoid false positives. The use of sentence boundaries is one such example. Without this functionality, "John Black" would also be matched in following text: "I have a friend named John. Black umbrellas are very popular."

Despite these precautions, false positives do occur. Another problem is that the Knowledgebase may contain multiple Entities named "John Black." To deal with cases like these, SEE uses a number of different mechanisms.

Since the Knowledgebase tracks relationships between Entities, a scoring mechanism uses this information to try to winnow down the possible matches by analyzing the connection between the Entities found in the text directly and through a single intermediary Entity not mentioned in the text. The aggressiveness of this reduction can be controlled and configured.

Another approach that further illustrates the utility of combining classification with metadata extraction is to use the domain of the document from classification to help determine the best Entities. In order for this to be effective, Entities must be associated with the classification taxonomy. Alternatively, EntityClasses may be associated with the domains and hence Entities that fall within a given class. In this way, when a document is assigned to the domain of politics, politicians are extracted preferentially. This technique reaches it limits, however, when the Knowledge contains multiple Entities named "John Black" who are politicians.

Conversely, Entities found in a corpus can be replaced with the corresponding EntityClasses in an attempt to replace the instance with an abstracted reference. This is analogous to "word concepts" in the LexisNexis classification system [6]. By so doing, a collection of training documents can be rendered more general, since occurrences of "John Black" would then match any politician once the substitution was made. Since the eMDExEnhancer works off the Knowledgebase, this information can be easily maintained.

The eRegexpEnhancer

As described in the previous section, named Entities are stored in a canonical representation mechanism known as the Knowledgebase. Dates, percentages, monetary amounts, phone numbers, email addresses and URLs are also potentially of interest. In order to allow this sort of knowledge to be exploited, the eRegexpEnhancer uses user-defined regular expressions to match arbitrary text.

Regular expressions are a popular and powerful way to perform pattern recognition. SEE's document normalization mechanism allows for regular expressions to span lines in the input text and ignore issues of encoding such as "&", """, and "<". Regular expressions have been used extensively in the past and are immensely popular in the programming community. By allowing the author of expressions to concentrate on defining the pattern without worrying about text format issues, their utility is expanded.

In this context, regular expressions can be thought of as "data scissors" that cut out meaningful pieces of documents, which can then be pasted into a semantically defined context. While in general this functionality is not unique [7], the ability to use regular expressions on a per category basis is.

The eRegexpEnhancer SEEM maps user-defined patterns to user-defined semantic types. Because it is a SEEM, this information is part of its configuration information. This

means that sets of regular expressions mapping to domain specific semantic types can be implemented quite readily. For each domain, a user of the system can assign semantic significance to any data for which they can define a regular expression.

So, if a document is categorized as a story about the stock market, for example, certain keywords such as "buys," "sell," "downgrade," and "sharp upturn" could be mapped to "FinancialEvent." When coupled with the ability to match named Entities, such as the names of companies, stock exchanges and ticker symbols, this simple mechanism can be powerful. Implementing a system like this to watch popular financial web sites or news feeds could be done with minimal effort, because the semantic infrastructure is already in place. The ability to use email addresses, phone numbers and URLs as semantic features may also be useful.

The ePhraseEnhancer

The last of the modules that create DocumentFeatures mapping directly to text regions is the ePhraseEnhancer module. Its purpose is to locate interesting strings of words in the document corresponding to phrases.

Since SEE does not currently provide NLP functionality, such as parts-of-speech tagging [8], ePhraseEnhancer identifies phrases as lists of capitalized words possibly separated by a short list of common words such as "of" and "the." Though this could be implemented as a regular expression, the document facilities of SEE make it much more efficient to have a specialized module. The phrases found are interesting for a number of reasons and are worth considering for a moment.

The names of people are frequently identified by the ePhraseEnhancer, and since this may potentially overlap with named Entity discovery, any phrases that match the same text region already matched by another DocumentFeature are discarded. Phrases have the lowest priority of all DocumentFeatures because they are so general and have such a low semantic value. In the cases where they are retained, phrases may still match a named Entity that was not picked up. This occurs in a number of scenarios. This module can pick up entities not in the Knowledgebase as well as alternative spellings and aliases for Entities that are. This can provide a handy mechanism for spotting shortcomings in the named Entity collection.

For a new installation starting from a fresh collection of documents with few Entities, the ePhraseEnhancer can be used to jump-start the Entity discovery process. Each file can be converted to a list of phrases and a TFIDF (term frequency / inverse document frequency) [9] value can be computed for each. By selecting the highest scoring phrases, probable Entities can be identified.

The eClassifyEnhancer

Automatic classification maps incoming documents to a domain in the taxonomy. In SEE, a committee of classifiers performs this classification. Numerous researchers have designed and implemented classification at a wide variety of academic [10] and commercial [11] institutions. After more than thirty years of research, there is still considerable debate and discussion over how this task can best be accomplished. In the document sets on which SEE's classification subsystem has been tested, varying from a test corpus such as Reuters-21578 and company internal portal with proprietary data and heterogeneous documents, there has been no single method which has always performed best. Instead of limiting classification to a single method, a committee approach has been taken which allows the text collection to dictate the classifiers used [12].

In addition to choosing a best-fitting collection of techniques, the classification committee is able to combine the results from a number of methods to arrive at a consensus about the probable category for a given document. As might be expected, this sort of approach is at its best when the classification techniques represented in the committee, function as differently as possible.

As the accuracy for the classifiers tends to vary for each document set, their results are weighted and combined as follows. First, the results for each classifier over all of the categories are scaled and normalized so that they fall within the range of zero to one. These per-classifier scores are then multiplied by a weight and summed to create a per-category score. Finally, this combined score is normalized with the score of one being assigned to the most probable category. These weights are determined at training time according to a variant of Larkey's and Croft's "Weighted Linear Combination" approach [13]. A genetic algorithm based approach is being developed to determine these weights as well [14].

Below are some results based on a subset of the Reuters-21578 text categorization test collection. Though the Lewis-Split (a well-known partitioning of the Reuters document set into training and test sets) used here details 144 separate categories, only a subset is shown here. The criterion used is that each category must contain at least 160 documents in its training set.

Table 1: The per-category, per-classifier accuracy rates for the Reuters document set where the number of training documents was at least 160 per category.

	acq	corn	crude	earn	grain	intst	money	ship	trade	wheat
Bay.1	64.95	0	7.98	2.02	1.39	1.52	1.69	1.15	0.85	1.45
Bay.2	99.58	0	3.72	15.35	1.39	1.52	2.25	0	0.85	1.45
HMM.1	97.22	89.29	82.45	95.13	81.94	79.55	87.08	77	92.31	92.75
HMM.2	97.77	85.71	78.72	98.62	85.42	63.64	90.45	47.1	87.18	92.75
HMM.3	97.22	85.71	78.72	98.62	85.42	62.12	88.76	47.1	88.03	91.3
Bay.3	99.58	0	3.72	70.4	1.39	1.52	2.25	0	0.85	1.45
HMM.4	0	8.93	1.6	0	6.25	1.52	2.25	5.75	2.56	5.8
Bay.5	0.14	73.21	3.19	0	33.33	6.06	3.93	9.2	5.13	34.78
KB	56.47	83.93	84.57	99.36	81.25	59.85	71.91	74.7	88.89	92.75
Bay.4	89.29	75	50	95.5	70.14	43.94	83.71	21.8	90.6	85.51
Combined	96.94	92.86	82.45	98.71	88.19	66.67	90.45	51.7	90.6	92.75

As can be seen in Table 1, the performance of the individual classifiers varies widely. Classifiers that produce poor results for a given set of documents are given lower weights in the combined score. The weights are computed by examining results on the training set. These weights are then used on the classifiers when the classification subsystem is tested.

Figure 3 shows results based on the Reuters-21578 text categorization test collection. In this test, we used a different threshold for the minimum number of documents per category in the training sets under the Lewis-Split, which produced the category counts at various thresholds. For each threshold, the categories meeting the minimum number of training documents served to train the classifiers, with tests then conducted on the corresponding test sets. The classification committee consistently outperformed the individual classifiers.

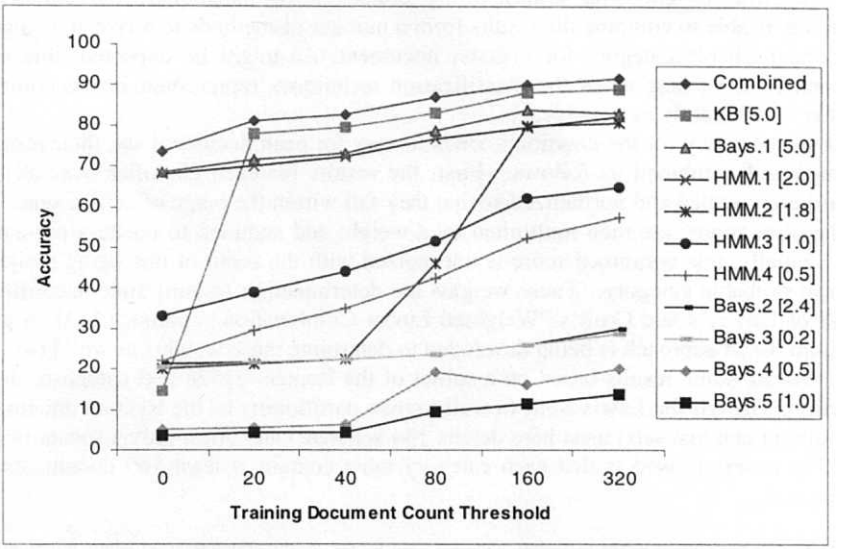


Figure 3: Accuracy rates for the individual classifiers on the Reuters set with various threshold values on the number of training documents required.

During this phase of operation, detailed information is available on a per-document basis. The document in the Reuters set labeled as 14859, reads:

> AMATIL PROPOSES TWO-FOR-FIVE BONUS SHARE ISSUE
> SYDNEY, April 8 - Amatil Ltd <AMAA.S> said it proposes to make a two-for-five bonus issue out of its revaluation reserve to shareholders registered May 26. Shareholders will be asked to approve the issue and an increase in authorised capital to 175 mln shares from 125 mln at a general meeting on May 1, it said in a statement. The new shares will rank for dividends declared after October 31. Amatil, in which B.A.T. Industries Plc <BTI.L> holds a 41 pct stake, said it does not expect to maintain its latest annual dividend rate of 29 cents a share on the enlarged capital.

All of the documents in the Lewis-Split have topic information associated with them. This particular document, like 75% of the Lewis-Split, is associated with a single category: earning. For this test, each document has been placed into a separate file.

Table 2: The per-category, per-classifier scores for a sample document along with the combined results.
/opt/shared/dox/reuters/limited_test/n_160/test/earn/reut.014.000858.sgm:

	bays1	bays2	bays3	bays4	bays5	hmm1	hmm2	hmm3	hmm4	kb	:	combined
acq	0.335	1.000	1.000	0.000	0.080	1.000	0.579	0.696	0.300	0.230	:	0.632
corn	0.016	0.000	0.000	1.000	0.000	0.000	0.027	0.059	0.940	0.153	:	0.035
crude	0.062	0.078	0.056	0.921	0.000	0.167	0.194	0.317	0.680	0.076	:	0.245
earn	1.000	0.315	0.281	0.684	0.919	0.655	1.000	1.000	0.000	1.000	:	1.000
grain	0.020	0.026	0.019	0.973	0.000	0.108	0.149	0.257	0.740	0.384	:	0.191
interest	0.003	0.026	0.018	0.973	0.000	0.095	0.095	0.178	0.819	0.153	:	0.128
money-fx	0.024	0.078	0.054	0.921	0.000	0.118	0.179	0.297	0.700	0.230	:	0.224
ship	0.047	0.026	0.017	0.973	0.000	0.038	0.000	0.000	1.000	0.230	:	0.000
trade	0.000	0.052	0.034	0.947	0.000	0.029	0.135	0.238	0.760	0.230	:	0.168
wheat	0.000	0.000	0.000	1.000	0.000	0.051	0.037	0.079	0.920	0.000	:	0.052
earn	earn	acq	acq	acq	earn	acq	earn	earn	earn	earn	:	earn

A result is considered to be correct if it matches any of the previously determined topics for a given document.

The first line is the name of the file being processed. The second line lists the names of the classifiers in the committee. The remaining lines give the name of the category and then the scores for each of the classifiers within the given category. The last line indicates the category predicted by the classifiers, with the first string representing the predetermined category and the last string giving the category determined by the committee. The guiding principles behind this format are ease of parsing and readability. When files are processed in batches, such as during the training and testing process, the entries are stored in a single file with blank lines separating the data.

Like SEEMs, the classifier committee is also modular. Classifiers can easily be configured to use stemming [15] and stop word lists. Generally speaking, however, these are enabled and disabled for all classifiers in the committee.

The string handling routines also contain a mechanism where certain words may be replaced with more generic versions. This list currently contains the following token or generalization: dollarAmountToken, percentageToken, floatingPointToken, monthToken, weekdayToken, yearNumberToken, timeToken, hrefToken, emailToken, largeAmountToken, numericToken, and nthToken.

This way the phrases "base run in the second inning" and "base run in the 7th inning" are both mapped to "base run in the nthToken inning." Thus training documents can be made more general to help capture the essential elements of the category. This also means that words like "70%," "1,000," and "1997" can be used more effectively.

Since the committee is designed to use a collection of classifiers, it also can take advantage of external classification technology. A wrapper has been written so that the rainbow classifier, running in server mode, can also take part in the classification process [16]. Thus, even if SEE's classification subsystem is not used, external classification technology can be seamlessly integrated into the system as a whole. This provides a way to leverage existing software infrastructures.

Semantic Annotation Example

The document from which this snippet was taken originally appeared on the web at money.cnn.com on August 22, 2002. The names of the companies and their stock symbols were hyperlinked in the paragraph starting "Tech stocks managed." SEE was not only able to uniquely identify entities, but also to find the relationships between them.

Here is the XML version of the enhanced text:

```
<EnhancedText>

<Classification domain="Business"/>

Blue-chip bonanza continues

Dow above 9,000 as
    <Entity id="494805" class="company">HP</Entity>,
    <Entity id="501293" class="company">Home Depot</Entity>
lead advance;
    <Entity id="851165" class="company">Microsoft</Entity> upgrade helps techs.

<Regexp type="date">August 22, 2002</Regexp>:
<Regexp type="time">11:44 AM EDT</Regexp>

By
    <Phrase>Alexandra Twin</Phrase>,
    <Phrase>CNN/Money Staff Writer</Phrase>

<Entity id="4764" class="city">New York</Entity> (CNN/Money) - An upgrade
```

```
of software leader <Entity id="851165" class="company">Microsoft</Entity>
and strength in blue chips including
    <Entity id="494805" class="company">Hewlett-Packard</Entity> and
    <Entity id="501293" class="company">Home Depot</Entity> were
among the factors pushing stocks higher at midday
<Regexp type="weekday">Thursday</Regexp>, with the
<Entity id="720500" class="financialIndex">Dow Jones industrial average</Entity>
spending time above the 9,000 level.

Around <Regexp type="time">11:40 a.m. ET</Regexp>, the
<Entity id="720500" class="financialIndex">Dow Jones industrial average</Entity>
gained 65.06 to 9,022.09, continuing a more than 1,300-point resurgence since
<Regexp type="date">July 23</Regexp>.
The <Entity id="505367" class="stockExchange">Nasdaq</Entity> composite
gained 9.12 to 1,418.37. The
<Entity id="211452" class="financialIndex">Standard &amp;amp; Poor's 500 index</Entity>
rose 9.61 to 958.97.

"Major indexes are up across the board," said
<Phrase>Peter Cardillo</Phrase>, director of
research at <Phrase>Global Partners Securities.</Phrase> "The
<Entity id="851165" class="company">Microsoft</Entity> upgrade certainly
helps, as does the absence of bad economic news. Also, oil prices are
retreating a little, the dollar is up and a little money is coming out
of Treasurys and into equities."

<Entity id="494805" class="company">Hewlett-Packard</Entity> (
    <Entity id="875349" class="tickerSymbol">HPQ</Entity>: up
    <Regexp type="money">$0.33</Regexp> to
    <Regexp type="money">$15.03</Regexp>,
    Research, Estimates
) said a
report shows its share of the printer market grew in the second quarter,
although another report showed that its share of the computer server
market declined
     in <Entity id="7852" class="continentRegion">Europe</Entity>,
    the <Entity id="7854" class="continentRegion">Middle East</Entity>
    and <Entity id="7848" class="continentRegion">Africa</Entity>.

<Entity id="501293" class="company">Home Depot</Entity> (
    <Entity id="511115" class="tickerSymbol">HD</Entity>: up
    <Regexp type="money">$1.07</Regexp> to
    <Regexp type="money">$33.75</Regexp>, Research, Estimates
) was up for the third straight day after topping fiscal second-quarter
earnings estimates on <Regexp type="weekday">Tuesday</Regexp>.

Tech stocks managed a turnaround.
    <Entity id="852744" class="techCategory">Software</Entity>
continued to rise after
    <Entity id="504523" class="company">Salomon Smith Barney</Entity>
upgraded No. 1 software maker
<Entity id="851165" class="company">Microsoft</Entity> (
    <Entity id="508968" class="tickerSymbol">MSFT</Entity>: up
    <Regexp type="money">$0.55</Regexp> to
    <Regexp type="money">$52.83</Regexp>, Research, Estimates
) to "outperform" from "neutral"
and raised its price target to
    <Regexp type="money">$59</Regexp> from
    <Regexp type="money">$56</Regexp>.

Business software makers
<Entity id="845065" class="company">Oracle</Entity> (
    <Entity id="508546" class="tickerSymbol">ORCL</Entity>: up
    <Regexp type="money">$0.18</Regexp> to
    <Regexp type="money">$10.94</Regexp>, Research, Estimates
),
<Entity id="496329" class="company">PeopleSoft</Entity> (
    <Entity id="508656" class="tickerSymbol">PSFT</Entity>: up
    <Regexp type="money">$1.17</Regexp> to
    <Regexp type="money">$20.67</Regexp>, Research, Estimates
) and
<Entity id="851574" class="company">BEA Systems</Entity> (
    <Entity id="508758" class="tickerSymbol">BEAS</Entity>: up
```

```
    <Regexp type="money">$0.28</Regexp> to
    <Regexp type="money">$7.12</Regexp>, Research, Estimates
) all rose in tandem.

</EnhancedText>
```

As can be seen in the XML above, each entity has a unique ID and a classification associated with it. Since the annotation is performed by a SEEM, the actual text of the tags can vary as needed. This can ease the integration of SEE into an existing system since it can be dropped in place and configured to yield custom input. In other situations, XSLT can be used to convert from one DTD to another or into HTML.

As shown in Figure 3, BEA Systems, Microsoft and PeopleSoft all engage in the "competes with" relationship with Oracle. When entities found within a document have relationships, we refer to the relationships as "direct relationships." Some of the direct relationships found in this example include: HPQ identifies Hewlett-Packard Co.; HD identifies The Home Depot; Inc.; MSFT identifies Microsoft Corp.; ORCL identifies Oracle Corp.; Salomon Smith Barney's headquarters is in New York City; and MSFT, ORCL, PSFT, BEAS are traded on Nasdaq.

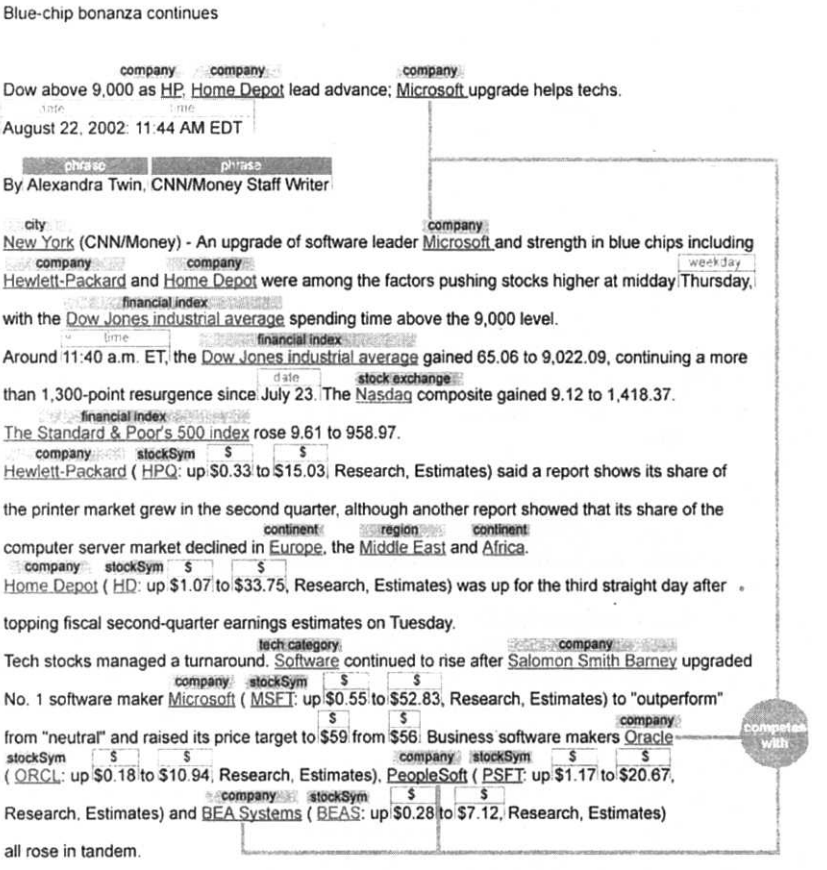


Figure 4: When SEE recognizes an entity, knowledge about its entity classification and semantic associations becomes available.

Not all of the associated entities for an entity found in the text will appear in the document. Often, the entities mentioned will have one or more relationships with another common entity. In this case, some examples include: HPQ and HD are traded on the NYSE; BEAS, MSFT, ORCL and PSFT are components of the Nasdaq 100 Index; Hewlett-Packard and PeopleSoft invest in Marimba, Inc., which competes with Microsoft; BEA, Hewlett-Packard, Microsoft and PeopleSoft compete with IBM, Sun Microsystems and Apple Computer.

The use of semantic associations allows entities not explicitly mentioned in the text to be inferred or linked to a document. This one-step-removed linking is referred to as "indirect relationships." The relationships that are retained are application specific and are completely customizable. Additionally, it is possible to traverse relationship chains to more than one level. It is possible to limit the identification of relationships between entities within a document, within a corpus across documents or allow indirect relationships by freely relating an entity in a document with any known entity in the SCORE Knowledgebase.

Indirect relationships provide a mechanism for producing value-added semantic metadata. Each entity in the Knowledgebase provides an opportunity for rich semantic associations. As an example, consider the following:

Oracle Corp.

Sector:	Computer Software and Services
Industry:	Database and File Management Software
Symbol:	ORCL
CEO:	Ellison, Lawrence J.
CFO:	Henley, Jeffrey O.
Headquartered in:	RedWood City, California, USA
Manufactured by:	8i Standard Edition, Application Server, etc.
Subsidiary of:	Liberate Technologies and OracleMobile
Competes with:	Agile, Ariba, BEA Systems, Informix, IBM, Microsoft, PeopleSoft and Sybase

This represents only a small sample of the sort of knowledge in the SCORE Knowledgebase. Here, the ability to extract from disparate resources can be seen clearly. The "Redwood City" listed for the "Headquartered in" relationship above, has the relationship "located within" to "California," which has the same relationship to the "United States of America." Each of the entities related to "Oracle" are also related to other entities radiating outward. Each of the binary relationships has a defined *directionality (some may be bi-directional)*. In this example, *Manufactured by* and *Subsidiary of* are marked as *right-to-left* and should be interpreted as "8i Standard Edition, Application Server, etc. are *manufactured by* Oracle" and "Liberate Technologies and OracleMobile are *subsidiaries of* Oracle." SEE can use these relationships to put entities within context.

When a document mentions "Redwood City," SEE can add "California," "USA," "North America." Thus, when a user looks for stories that occur in the United States or California, a document containing "Redwood City" can be returned, even though the more generalized location is not explicitly mentioned. This is one of the capabilities a keyword-based search cannot provide. The interesting thing about this example is just how much information can be found in a typical document. By placing this information within context, the information implicit in the text is revealed and can then be linked with other sources of content.

9. Related Work

SEE shares objectives with projects such as CARMEN [17]. However, its document processing is significantly more comprehensive as it deals with heterogeneous types of data and broader analysis of text. Its metadata extraction capabilities are significantly more advanced than primarily statistical techniques reported in CARMEN [17].

Riloff and Jones [18] describe a technique of extracting metadata based on a bootstrapping technique in which an initial seed of entities is used to discover linguistic patterns or triggers that are, in turn, used to discover additional entities. The approach used by SEE's eMDExEnhancer is based instead upon the use of a relevant ontology (and more specifically the Knowledgebase of SCORE). Rather than trying to build patterns, internal and external knowledge sources are used to create a comprehensive list of all possible entities of interest in the domain. SEE's use of classification also allows for multiple domains to be fed off of the same Knowledgebase by targeting specific groups of entities based on the results of automatic classification. This means that SEE will be able to find entities that may not have appeared in the collection of training documents.

Nymble [19] uses a tagged corpus to train an HMM variant to automate data extraction. Like many information retrieval oriented systems, it identifies entities by type, but not *specific* entities. While such a system can discover "IBM" as a company in a document, it does not have the ability to make the connections to "Mainframes," "Microchannel Bus," and "IBM Global Services" that SEE provides (along with the ability to identify and annotate with relevant relationships if modeled as part of the ontology, specifically the WorldModel of SCORE). The ability to add information contextually by exploiting semantic relationships simply not available.

Systems that rely on discovering patterns in training text without canonical lists of entities [20] have achieved good results. However, they make the assumption that incoming document will closely match the wording and entity sets of the documents in the training collection. By leveraging the knowledge available from the World Wide Web, internal databases and human experts, this assumption can be avoided. Keeping the Knowledgebase up-to-date, maintaining both the pool of entities and their interrelations, allows SEE not only to identify the occurrence of entities, but to identify entities exactly: not just a TV show called "Cowboy Bebop," but *the* TV show "Cowboy Bebop." The correct identification of individual entities allows the power of semantic association to be brought to bear.

Semantic metadata annotation is an important area of focus in the Semantic Web research. SEE supports significantly more automation than the current semi-automatic annotation tools in academic research (such as OntoMat-Annotizer, [21]) or available as a product (such as Ontoannotate[2]).

Conclusions

SEE is able use its highly configurable architecture to support document processing at a per-document level for format parsing, word parsing, and phrase/sentence parsing. It also puts documents into context through automated classification methods. Once the domain of discourse is known, metadata extraction techniques can be customized on a per-category basis. This targeting effectively reduces the range of possible types of information that need to be analyzed for a given document. Knowing when to look for CEOs and when to look for baseball players improves accuracy. Matching regular expressions for one category only increases throughput. By restricting the scope of semantic metadata, SEE

[2] http://www.ontoprise.de/com/start_products.htm

makes it possible to deal with the world one domain at a time. Through the identification of the unique entities in a document, SEE is able to harness the knowledge available from disparate sources and use it to add meaning and relevance to text. SEE is also unique in its ability to find contextually relevant relationships. The scope of relevant knowledge used and the documents across which relationships are identified can be controlled. We believe SEE takes advanced heterogeneous document processing to a semantic level with highly automated metadata extraction and semantic annotation capabilities. Semantic annotation is likely to be one of the first critical scalability challenge that the Semantic Web will face. Doing so automatically with high quality metadata is important, and this work presents a step in addressing that challenge.

Acknowledgements

We would like to thank our colleagues who helped make this work possible. Special thanks go to Clate Sander for assistance with presentation.

References

[1] N. Collier, "Machine Learning for Information from XML-markup on the Semantic Web," 2000.

[2] A. Maedche, S. Staab, "Mining Ontologies from Text," 12th International Workshop on Knowledge Engineering and Knowledge Management (EKAW 2000), October 2000.

[3] P. Clerkin, P. Cunningham, C. Hayes, "Ontology Discovery for the Semantic Web Using Hierarchical Clustering", Semantic Web Mining Workshop at ECML/PKDD-2001, September 3, 2001, Freiburg, Germany.

[4] A. Sheth, D. Avant, and C. Bertram, "System and Method for Creating Semantic Web and Its Applications in Browsing, Searching, Profiling, Personalization and Advertisement," U.S. Patent #6,311,194, 30 Oct. 2001.

[5] A. Sheth, C. Bertram, D. Avant, B. Hammond, K. Kochut, and Y. Warke, "Semantic Content Management for Enterprises and the Web," IEEE Internet Computing, July/August 2002, pp. 80-87.

[6] Mark Wasson, "Classification Technology at LexisNexis," SIGIR 2001Workshop on Operational Text Classification

[7] C. Grover, C. Matheson, A. Mikheev, and M. Moens, "LT TTT - A Flexible Tokenisation Tool", in Proceedings of the Second Language Resources and Evaluation Conference, 31 May-2 June 2000, Athens, Greece.

[8] E. Brill, "Some Advances in Transformation-Based Part of Speech Tagging," National Conference on Artificial Intelligence, 1994.

[9] G. Salton, "Developments in Automatic Text Retrieval Science," Vol. 253, pages 974-979, 1991.

[10] F. Sebastiani, "Machine Learning in Automated Text Categorization," *ACM Computing Surveys*, vol. 34, no. 1, 2002, pp. 1-47.

[11] C. Adams, "Word Wranglers: Automatic Classification Tools Transform Enterprise Documents from 'Bags of Words' to Knowledge Resources." Intelligent KM, January 2001.

[12] R. Liere and P. Tadepelli, "Active Learning with Committees for Text Categorization," *Proc. 14th Conf. Am. Assoc. Artificial Intelligence*, AAAI Press, Menlo Park, Calif., 1997, pp. 591-596.

[13] L.S. Larkey and W.B. Croft, "Combining classifiers in text categorization," Proceedings of SIGIR-96, 19-th ACM International Conference on Research and Development in Information Retrieval, 1996, pp.289-297.

[14] Z. Byoung-Tak and J. Je-Gun, "Building Optimal Committees of Genetic Programs," Parallel Problem Solving from Nature - PPSN VI 2000

[15] M. Porter, "An algorithm for suffix stripping," Technical Report 14, Program, 1980. http://www.muscat.co.uk/~martin/stem.html.

[16] A. McCallum, "Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering," http://www.cs.cmu.edu/mccallum/bow/, 1996.

[17] J. Krause and J. Marx, "Vocabulary Switching and Automatic Metadata Extraction or How to Get Useful Information from a Digital Library," Proceedings of the First DELOS Network of Excellence Workshop on Information Seeking, Searching and Querying in Digital Libraries, Zurich, Switzerland, December, 2000.

[18] E. Riloff and R. Jones, "Learning Dictionaries for Information Extraction by Multi-Level Bootstrapping", Proceedings of the Sixteenth National Conference on Artificial Intelligence, 1999.

[19] D. Bikel, S. Miller, R. Shwartz and R. Weischedel, "Nymble: a high-performance learning name-finder," Proceedings of ANLP-97.

[20] A. Mikheev, M. Moens and C. Grover. "Named Entity recognition without gazetteers," Proceedings of EACL, Bergen, Norway, 1999.

[21] S. Handschuh and S. Staab, "Authoring and Annotation of Web Pages in CREAM," WWW2002, May 7-11, 2002, Honolulu, Hawaii, USA.

Real World Semantic Web Applications
V. Kashyap and L. Shklar (Eds.)
IOS Press, 2002

From Conceptual Models to Constrained Web Forms

Marlon Dumas, Lachlan Aldred, Arthur H.M. ter Hofstede
Centre for Information Technology Innovation
Queensland University of Technology
GPO Box 2434, Brisbane QLD 4001, Australia
{m.dumas, l.aldred, a.terhofstede}@qut.edu.au

Abstract Constraints expressed within document definitions, data models, ontologies, and domain models, are a key constituent of the Semantic Web. Indeed, Semantic Web applications will rely on these constraints when reasoning about Web contents. This paper deals with the enforcement by Web interfaces, of constraints expressed at the conceptual level. Specifically, the paper presents an approach to mark up a conceptual model in order to generate a Web form capable of enforcing the constraints captured in the model. The feasibility and practicality of this approach have been validated through the development of a system that generates XForms code from ORM conceptual schemas.

1 Introduction

Rules in general, and constraints in particular, play a key role within document definitions, data models, ontologies, and domain models, which are fundamental building blocks of the Semantic Web. The enforcement of constraints is thus crucial for the realisation of the Semantic Web. Indeed, Semantic Web applications will rely on these constraints when processing and reasoning about contents in order to perform tasks on behalf of their users. This chapter deals with the issue of enforcing constraints over Web contents through Web forms.

At present, Web forms are commonly developed in an ad-hoc way, either directly in a markup language (e.g. HTML/Javascript and more recently XForms [16]) or with the assistance of a graphical Web page editor. As a result, the constraints over the underlying application domain are manually translated into constraints over the newly created Web form. We foresee that in the next generation of Web applications development tools, this translation process will be automated, in such a way that the intrinsic structure and semantics of Web contents will be used to frame the interaction of the applications with their users. With this in mind, we describe an approach to attach presentation markups to conceptual models capturing constraints over Web data, in order to generate Web forms capable of interactively enforcing the constraints captured in these models.

An application of this automated generation of Web forms can be found in E-Commerce Web sites. Indeed, an E-Commerce Web site developer can mark up a conceptual model capturing various product categories, and utilise the proposed generator to produce Web forms for the front-end of the system. The sellers can then use these forms to enter the details of their specific offerings, and the data collected by these forms can be used to create product descriptions. The same conceptual model can also be marked up in order to build forms

through which buyers (either directly or through a buying agent) can designate the products they are interested in.

The rest of the chapter is organised as follows. We start by overviewing the conceptual modelling language used to represent constraints over Web contents, and by introducing an example conceptual model (Section 2). We then present the approach to markup conceptual models in order to generate Web forms (Section 3). Next, we describe an implementation of this markup and generation approach (Section 4). This description is followed by a review of related work (Section 5) and some concluding remarks (Section 6).

2 Constraints in conceptual schemas

2.1 Choice of a conceptual modelling language

Constraints over Web contents can be represented using at least two families of languages: knowledge representation languages (e.g. DAML+OIL [4]) and conceptual modelling languages (e.g. UML [14] and ORM [10]). Knowledge representation languages typically support the concepts of class, property, is-a relationship, cardinality constraints, and allow designers to express general-purpose rules in a first-order logic language. Conceptual modelling languages also support the concepts of class, association, and subtyping (or inheritance), but in addition to this, they provide a large collection of *constraint types* (e.g. cardinality constraints, mandatory association constraints, set constraints). Conceptual modelling languages allow designers to express general-purpose constraints in a logic-based language such as UML's OCL (Object Constraint Language) or FORML (Formal ORM Language). However, these general constraints are treated separately from those expressed using constraint types.

For the purposes of generating Web forms, which typically encode very specific constraint types (e.g. mandatory fields, correlated fields, etc.), the use of conceptual modelling languages seems more appropriate. Indeed, if a knowledge representation language was used as a starting point to generate Web forms, it would not be possible to encode in the generated forms many of the rules that can be represented in the source language.

Having chosen to represent the structure and constraints of Web contents using a conceptual modelling language, there is a further choice between several alternatives. Previous research in the area of the Semantic Web has considered the use of UML class diagrams [5] and ORM [6] for modelling Web contents (especially business rules and ontologies). Although UML is certainly the most widely used of these two modelling languages, ORM has the advantage of providing a richer set of constraint types (i.e. constraints with a predefined structure such as cardinality constraints or set constraints over associations). Indeed, ORM has more than 10 types of constraints, whereas UML constraint types are essentially limited to cardinality constraints and exclusion constraints over inheritance links[1] [10]. Hence, a technique that would be able to translate all constraint types in ORM into constraints over Web forms, could easily be adapted to deal with UML class diagrams. For this reason, we have chosen to base the presentation of the approach on ORM.

To make the chapter self-contained, we provide an overview of the ORM language. This overview is not complete, as it is tailored to support the discussions in the rest of the paper. For full details of the ORM notation, the reader is referred to [10].

[1] In UML, arbitrarily complex constraints can be expressed using OCL. However, in this chapter we do not consider the enforcement of arbitrary constraints expressed as first-order logic expressions.

2.2 Basic constructs of ORM

The basic concepts of ORM are those of *entity* and *fact*. Entities are the real or abstract things being modelled (e.g. the objects of an application, or the terms of an ontology), while facts are the statements that are made about entities (e.g. their relationships). A fact relates two or more entities, each of them playing a different *role* in the fact. For example, "drives(john, truck122)" is a fact relating the entities "john" and "truck122". Entities are grouped into *entity types* and facts are grouped into *fact types*. A fact type is composed of one or more *roles*. For example, a fact type "drives" would have two co-roles: one of them played by the entity type "Person", and another one played by the entity type "Vehicle". Unlike other modelling techniques, ORM makes no distinction between the concepts of "attribute" and "association". Instead, all the relationships between things of the Universe of Discourse are modelled through fact type:

Each entity type has an *identification scheme* used to identify entities of that type. Identification schemes can be simple or complex. A simple identification scheme is made up of a single *value type* (e.g. set of literals). A complex identification scheme combines several types. Examples of simple identification schemes include the set of names of car makes (MakeName) and the set of numbers (nr)+. An example of a complex identification scheme is the combination of a type "Street Number", a type "Street Name", and a type "Suburb Name", which together identify an address.

At the instance level, an entity type is a set of entities, while a fact type is an N-ary relation (also called the *population* of the fact type).

Graphically, an entity type is denoted by a named ellipse. If the entity type has a simple identification scheme, this appears parenthesized below the name of the entity type. Fact types are denoted by named sequences of contiguous boxes: one box per role in the fact type. For example, a binary fact type is denoted as two contiguous boxes. Each box (role) is linked through an edge to the ellipse (entity type) playing that role.

An example ORM diagram is given in Figure 1. This diagram contains the entity types Car, Model, Make, and Year among others. It also features a number of fact types relating these entity types (e.g. a fact type relating Car to Model, another relating Car to Make, etc.).

The special fact type "subtype of", denoted by a directed arrow going from the subtype to the supertype, captures a notion of inheritance. If an entity type Sub is a subtype of an entity type Sup, then any entity of Sub is also an entity of Sup, and consequently, all the fact types involving Sup also involve Sub. Moreover Sub should have a subtyping condition associated to it, typically expressed in terms of the fact types in which Sup is involved. All the instances of Sup which satisfy this condition are in Sub. For example, Figure 2 shows a fragment of a diagram in which the entity type type Car is declared to be a subtype of the entity type Item. The associated subtyping condition states that a car is an item whose category is "Car". The entity type Car plays the role "has price" (i.e. a car has a price).

2.3 Constraint types in ORM

The following types of constraints (among others) can be captured in an ORM diagram:

- Value constraint: enumerates in extension or through ranges, the values that a type can take. In Figure 1, there is a range value constraint under the type Year (and another one under the type NumDoors), stating that the schema only deals with years greater than or

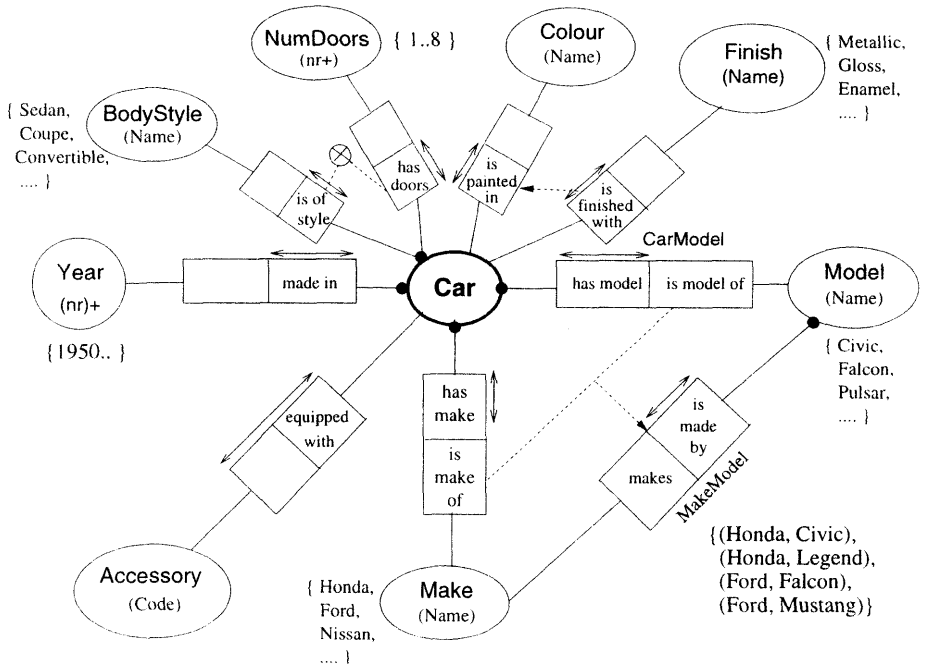


Figure 1: Partially populated ORM schema.

equal to 1950. There are also enumerated value constraints associated to the types Model, Make, Finish, and BodyStyle.

- Mandatory role constraint: states that all the entities of a given type must play a role (or combination of roles). In the absence of such a constraint, some entities might not play the role (or combination of roles). The fact that a role is mandatory is denoted by a filled circle at the extremity of the line connecting the role to the entity type. In Figure 1, the roles "has model", "has make" and "made in" are mandatory. On the other hand, the role "is painted with" is not mandatory, meaning that for some cars, we might not know the colour. Also, the combination of roles "has doors" and "is of style" is mandatory, meaning that an entity of type Car must play at least one of these two roles.

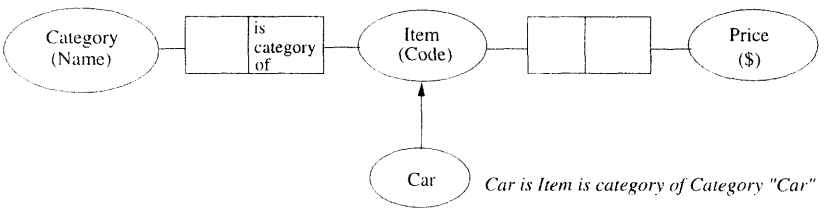


Figure 2: Fragment of an ORM schema illustrating a subtyping.

- Uniqueness constraint: when applied to a role R (or a combination of roles) of a fact type F, this constraint states that a given entity (or combination of entities) cannot play role R more than once. In other words, an entity can only be related to one other entity (or combination of entities) through the fact type F.
  In the case of binary fact types, this constraint type serves to state whether a binary fact type captures a 1:1, a 1:N or a N:M relationship. In Figure 1, the fact type CarModel has a uniqueness constraint on the role "has model" (indicated by a double arrow on top of the role) meaning that a car has only one model. There is no uniqueness constraint on the role "is model of", so several cars can have the same model. Hence, the fact type CarModel captures a 1:N relationship. The CarAccessories fact type on the other hand models a N:M relationship, since the uniqueness constraint covers both roles of the fact type, meaning that none of the roles individually forms a key of the fact type.

- Occurrence frequency (N..M): when attached to a single role R, states that if an entity E plays role R, then the number of times that E plays R is between N and M. Notice that the uniqueness constraint type is in fact a particular case of the occurrence frequency constraint type, where N = 1 and M = 1. For example, if we wanted to model the fact that a car has at most 10 accessories, we would place an occurrence frequency constraint of the form 1..10 over the role "equipped with" in Figure 1.

- Set exclusion: in the general case, this constraint states that the sets to which it applies, must be disjoint. When applied to roles, it states that an entity of a type can play only one among several roles. For example, the set exclusion constraint between the roles "is of style" and "has doors" in Figure 1, indicates that in this conceptual schema, a car cannot play these two roles simultaneously, that is, it cannot have a "number of doors" and a "body style" at the same time.

- Set inclusion: states that one set is included in another. When applied to roles, it captures the fact that all the entities (or combination of entities) playing a role (or a combination of roles) must also play another role (or combination of roles). An inclusion constraint is represented by a dotted directed arrow from the subset to the superset. For example, in Figure 1, there is an inclusion constraint between the roles "is finished with" and "is painted in", meaning that if the finish of a car is known, the colour should also be known. Similarly, there is also a set inclusion between the combination of the roles "is make of" and "is model of" on the one hand, and the roles "makes" and "is made by" on the other, meaning that if a car is of make MK and of model ML, then model ML must be related to MK through the fact type MakeModel.

When describing Web contents, whether Web data models, domain models, or ontologies, it is often important to consider not only the structure of the Universe of Discourse (its schema) but also its population. In these cases, a conceptual schema can be augmented with a population, leading to what is called a (partially or totally) *populated conceptual schema*. The population of an ORM schema is composed of the populations of the fact types of the schema. As mentioned above, the population of a fact type is a set of tuples. In the example of Figure 1, the fact type MakeModel is populated by attaching to it a set of pairs composed of a name of a model and a name of a make. The population of a fact type need not be represented in the diagram: it can be stored separately.

3 Marking up conceptual models

In this section, we sketch a method for marking up partially or totally populated conceptual schemas in order to generate Web forms encoded in XForms. To make the chapter self-contained, we provide a brief overview of XForms prior to describing the markup and generation processes.

3.1 Overview of XForms

Currently, Web forms are being encoded as a combination of HTML tags and embedded routines written in scripting languages such as Javascript. The HTML tags capture the structure and rendering of the form, while the routines encode the reactive aspects, such as performing range checks, or correlating different parts of a form. The resulting Web form code is hard to read, to maintain, and to evolve, due to the fact that the purpose of the form (i.e. collecting data that conforms to a given data model and/or ontology), is mixed with its presentation (i.e. the rendering of the form). In addition, form control elements in HTML are quite basic, since they were originally designed to capture a restricted set of user interactions.

To address these limitations, the W3C HTML Working Group has designed XForms [16] as a successor of HTML forms. XForms is an XML-compliant language for describing Web forms in a way that separates the purpose from the presentation. The framework includes a language for specifying device-independent forms and a language for representing the data gathered from a form.

The data gathered from a form (the *instance data*) is formatted as an XML document conforming to the XML schema attached to the form description. The initial value of the instance data is provided within the form description, and it is incrementally updated by the XForms processor everytime that the user interacts with the form. In addition to the constraints over the instance data expressed in its XML schema, other constraints can be attached (bound) to the instance data using XPath expressions. Three of these additional constraints are used in the rest of the paper: "relevant", "required", and "IsValid". A "relevant" constraint states that a given instance data element is relevant only when a given XPath expression evaluates to true; otherwise, the data element is not relevant and should be hidden or made unavailable for modification. The "required" constraint states that a given data element is required only if a given XPath expression evaluates to true: when a form is submitted, if one of its data elements is required, and no value is provided for it, then the XForms processor issues an error message. For example, by attaching an appropriate XPath expression to a form control element (e.g. Body Style), it is possible to declare that this form control element is required if the user has not filled in another form control present within the same form (e.g. Number of Doors). Finally, an "IsValid" constraint states that a given data instance is only valid if a given XPath expression is true. This can be used for example to perform range checks.

XForms offers 7 types of form control elements for entering data:

1. Input: for entering strings.
2. Secret: for entering e.g. passwords.
3. TextArea: for entering texts.
4. Upload: for uploading files.

5. Range: for selecting a number within an interval.
6. SelectOne: for selecting one among a given set of items. Can be implemented for example as a pull-down menu.
7. SelectMany: for selecting many items among a given set. Can be implemented for example as a group of checkboxes allowing multiple selections.

Regardless of its type, each form control element has several common attributes, including its identifier (internal name), its caption, its description (also called "hint"), and its documentation.

In addition, XForms offers 3 composition operators over form control elements: "group", "switch" and "repeat". The "group" operator simply clusters several form controls together, so that they appear within the same block in the user interface. The "switch" operator allows the forms developer to conditionally compose several form controls so that some of the form controls only appear when some condition is satisfied, while others appear if another disjoint condition is satisfied. Finally, the "repeat" operator states that a (determined or undertermined) number of instances of a given control element should appear in the user interface. For example, the "repeat" operator applied to an input field, yields a composite form control that allows several data items to be entered (one per input field).

At present, XForms are supported by several prototype systems. Chiba[2] for example translates XForms code into plain HTML, and implements the dynamic aspects of a form through server-side processing of the (partial or final) submissions of the form. The main drawback of this server-oriented approach is that the client needs to interact with the server every time that an event occurs (i.e. the user wishes to add an item to a "repeat" composite form control). Other implementations such as TrustForms[3] and XSmiles[4] are client-side: they are essentially XForms-enabled browsers. In these implementations, the XForms code is interpreted by the browser, and the server-side processing is only required during the final submission of the form. Although these tools are still in their early stages of development, it is expected that full-fledged, mature implementations of XForms will be available soon [16].

3.2 Generating forms for creating an instance of a type

In this section, we describe the process of marking up a conceptual schema to generate a Web form for creating an entity O of a given type T. This same form can also be used to edit (i.e. to modify) an object of an entity type, except that in the case of object modification, some of the form controls are already filled in (i.e. they contain previous values) when presented to the user, whereas in the case of object creation, the fields do not necessarily have previous values: they are either empty or filled in with default values.

To simplify the presentation, we assume that all fact types are binary. Ternary and higher arity fact types can be marked up in the same way as binary ones, except that they are mapped to combinations of form control elements (e.g., tables of input fields), instead of individual ones as below.

[2] `http://sourceforge.net/projects/chiba`
[3] `http://trustform.comsquare.co.kr`
[4] `http://www.x-smiles.org`

In the first step of the markup process, a subset of the set of roles in which T or any of its subtypes participates, are selected as those roles for which a form control element will be included in the generated form. The reason why roles belonging to the subtypes of T are allowed to be selected, is to enable dynamic object classification: the object to be created (or edited) through the generated form, can belong to any of the subtypes of T, depending on whether it satisfies the subtyping conditions associated to these subtypes.

Next, for each role R selected in the previous step, the following data items should be provided by the form developer:

1. The caption of the form control generated for role R. For example, the caption for the role "has make" in Figure 1 could be "Make".

2. A description (or hint) of the form control generated for role R. For example, the description of a SelectOne form control corresponding to the role "has make" could be *"Select a make of car"*. This data item is optional.

3. A documentation (i.e. a help text) of the form control generated for the role R. For example, the documentation of the role "has make" could be: *"The make of a car is the brand given to it by its manufacturer. Examples of makes are Honda and Ford"*.

4. Whether the entity O′ to which O is related through the fact type containing R is to be designated:

   - By an identifier (e.g. an agreed-upon label or an URI)
   - By means of a sub-form

   These two cases are respectively called *designation by identifier* and *designation by sub-form*.

   In the example of Figure 1, the developer will use designation by identifier for all the fact types involving the type Car, since according to the conceptual schema, there is no other means of designating (e.g.) a make, a model or a year, other than by an identifier. Sub-forms would be useful for example in the case of a conceptual schema involving an entity type Hotel related to a type Address. In this setting, a form for designating an object of the type Hotel is likely to involve a sub-form for designating the address of the hotel. This sub-form would contain an input element for each of the fact types involving the type Address (e.g. street number, street name, city).

5. In the case of designation by identifier, the type of form control to be generated for role R (Input, SelectOne, SelectMany, Range, etc.).

   In the working example, the fact types CarModel and CarMake can be marked up so that a SelectOne form control is used to represent them, while the fact type CarYear can be mapped to an "Input" form control. More generally, if the entity type playing the co-role of R has an enumerated value constraint attached to it (i.e. if we know all the possible makes of cars), it is possible to map this role into a SelectOne form control element. Otherwise, the role can only be mapped into an input element, text area, or a range element (in the case of numeric value types).

For a role R designated by sub-form, the marking process is recursively repeated, this time taking as starting point the entity type that plays the co-role of R. In the working example (i.e. creating an object of type Car), no designation by sub-form occurs, so the process is stopped at this point. In the example of marking up a conceptual model to create an entity of type Hotel (as discussed above), the markup process would need to be repeated, taking as starting point the type Address (to which the entity type Hotel is related).

3.3 Forms for designating an instance of an entity type

The process of marking up a conceptual model to generate a form for designating an entity O of type T, is similar to the markup process described above in the context of object creation. First, a subset of the fact types (or roles) in which T participates are selected as those roles for which a form control element will be included the generated form. These fact types must be chosen in such a way that specifying the entity to which O is related through each of these fact types, leads to an unambiguous designation of O. In other words, the set of selected fact types must contain a complex identification scheme of type T (see Section 2.2). For example, in the context of designating an entity of type Person, the set of selected roles would be something like { FirstName, LastName, DateOfBirth }, since these three items are enough to unambiguously designate a person in the general case.

The rest of the process is identical to the above one: for each of the fact types (or roles) selected in the previous step, the form developer enters several data items establishing the way in which these fact types are to be presented in the generated Web form.

3.4 Translation of the conceptual model constraints

The uniqueness constraints contained in a conceptual schema are enforced during the markup process, by allowing the developer to select only those kinds of form controls that are compatible with the cardinality of a selected role. For example, if the cardinality of a role is "one", then the developer can only select a form control that allows to enter a single value (e.g. SelectOne) as the form control element to be associated to that role.

Other constaints captured in an ORM schema are taken into account during the generation of forms. For example, value constraints in an ORM schema are translated into range checks in the code of the generated form. In the example of Figure 1, a range check will be performed by the generated form to ensure that the value entered for the year is greater than or equal to 1950, thereby capturing the value constraint appearing under the entity type Year.

Mandatory role constraints on the other hand are used to determine whether a form control is required or not (i.e. whether entering a value in this form control is required before submitting a form). For example, in the working example, the form controls corresponding to the roles "has model", "has make" and "made in" will be required before form submission (i.e. the user must enter a value in these form controls), since these roles have mandatory constraints. Also, the form controls corresponding to the roles "is of style" and "has doors" will be *jointly required*, meaning that the user will have to enter a value in at least one of these fields. Furthermore, the set exclusion constraint between these two roles entails that if the user fills in the form control corresponding to one of these roles, (s)he may not fill the form control corresponding to the other role. It follows that the user must fill in one and only one of the form control elements corresponding to these two roles ("is of style" and "has doors").

ORM constraint	Web form feature
Simple mandatory role (i.e. a selected role is mandatory)	Mandatory form control: if the form control is multi-valued, at least one value must be entered.
Complex mandatory role (i.e. a combination of selected roles is mandatory)	Mandatory combination of form controls: it is compulsory to fill at least one of the form controls associated to the roles participating in this constraint
Uniqueness constraint over a selected role	The form control corresponding to this role should be single-valued (e.g. a text input or a SelectOne form control). If there is no uniqueness constraint over a selected role, then the control element corresponding to this role should be multi-valued (e.g. a SelectMany or a repeated input field).
Occurrence frequency N..M over a selected role ($M \geq 2$)	The form control corresponding to this role should be multi-valued (e.g. a SelectMany or a repeated input field) but the number of values that can be entered through this control element should be between N and M.
Value constraints over a type that plays the co-role of a selected role.	Restricts the set of values that can be entered over the form control corresponding to the selected role.
Exclusion constraints between several selected roles	At most one of the form control elements corresponding to these roles can be filled in.
Subset constraint between two selected roles of different associations	If the form control corresponding to the subset role is filled in, the form control corresponding to the superset role has to be filled in as well.
Equality constraint between two selected roles of different fact types	If the form control corresponding to one of the roles is filled in, the form control corresponding to the other role must be filled in as well.
Subtyping condition ϕ over a subtype S of the entity type T	The form controls associated to the fact types in which S (but not T) participates become irrelevant (and are therefore hidden), whenever ϕ evaluates to false.
Join between two selected roles L1 and L2 and subset constraint between the resulting join, and a fact type linking the types playing the co-roles of L1 and L2.	Correlated form controls: when a value is selected in the form control corresponding to L1, the form control corresponding to L2 is updated to reflect this fact. For example, when a car make is selected in one of the form controls, the set of car models corresponding to that make appear in the other form control.

Table 1: Mapping of ORM constraints to Web form features

Similarly, the set inclusion constraint between the roles "is finished with" and "is painted in" will ensure that if the user fills in the form control corresponding to the role "is finished in", (s)he must also fill the form control corresponding to "is painted with" (i.e. if the finish is specified , the colour must also be specified).

Finally, the inclusion constraint beetween the join of the roles "is make of" and "is model of", and the fact type MakeModel, can be enforced in the generated form, by correlating the form controls corresponding to the roles "has make" and "has model". Assuming that both of these form controls are of type SelectOne, these menus will be correlated as follows: if the user selects a given make of car (in the menu corresponding to the role "has make"), only the models of the selected make will appear as options in the menu corresponding to the role "has model". The options of these two correlated menus are derived from the population of the fact type MakeModel.

More generally, Table 1 describes mappings from ORM constraints involving a selected role, to features that can be encoded in a Web form to enforce these constraints. This table is not exhaustive: it does not consider any constraint whose translation into a Web form feature, would require that the entire population of a fact type is included in the Web form description. For example, consider the case of a conceptual schema with an entity type "Person" and an entity type "Social Security Number", linked by a fact type "PersonNumber" modelling a 1:1 association (i.e. a person has only one number and no two persons can have the same number). One can imagine that a Web form for creating a new entity of type Person, would be able to check that the social security number entered for a new person, is not already taken by another existing person, by looking at the population of the fact type "PersonNumber". However, in order to perform this validation client-side, the Web form description would need to contain all the social security numbers of all the persons appearing in the fact type's population. In general, this population is too large to be sent through the network together with the Web form. Hence, the constraint checking should be rather done on the server. Such a server-side validation is beyond the scope of this paper[5].

The constraint mappings in Figure 1 apply both in the case of generating a form for creating an entity, and in that for designating an entity. In the specific case of a form for designating an entity, the populations of fact types are further exploited during Web form generation. Specifically, if a populated fact type is mapped into a SelectMany form control, the options that appear in this menu are derived from the population of this fact type. With respect to the example of Figure 1, if the population of the role "is make of" is { Honda, Ford, Toyota }, the SelectMany menu corresponding to the make will contain these values as its options. Similarly, in the case of a text field, the population of the corresponding role is used to check that the term entered by the user in this field actually exists in the model.

4 Implementation

4.1 System overview

An overview of the Web forms generation system is given in Figure 3, where the circles denote system modules, the boxes denote inputs and outputs of these modules, and the dashed

[5]The last constraint in the table does require to include an entire population of a fact type within the generated XForms document. We include it in the table because we believe that in the presence of this kind of constraint, the population of the involved fact type is likely to be small enough to be included in a Web form.

lines denote interactions between the system and the users.

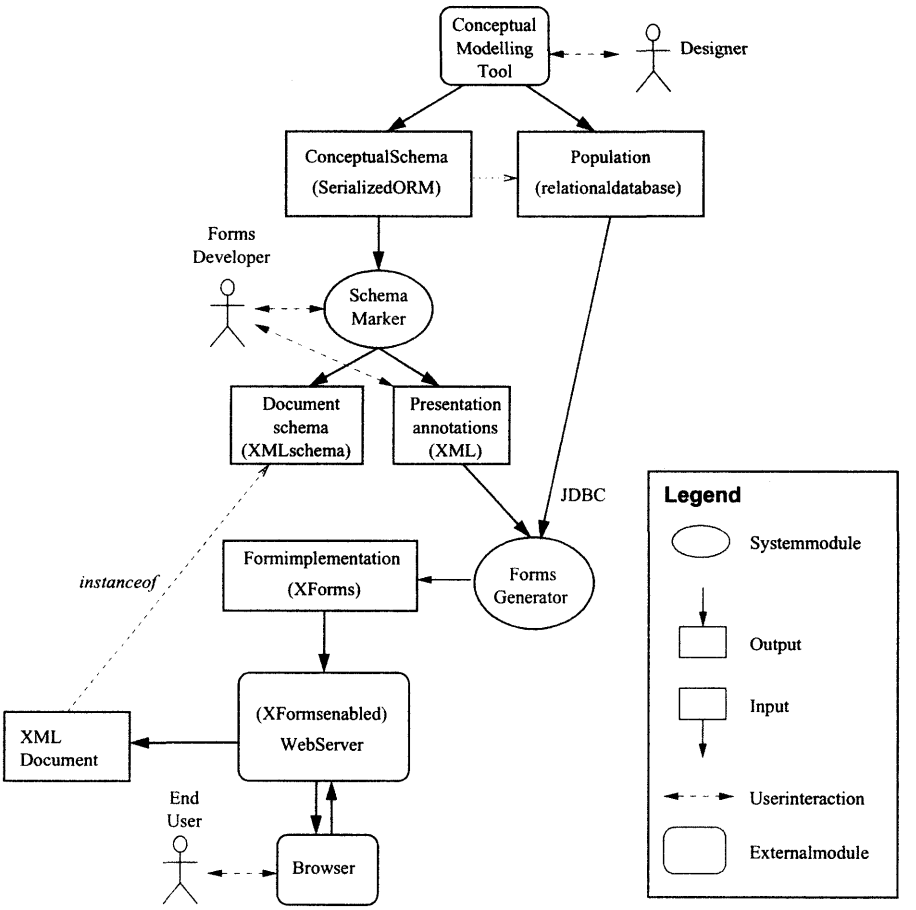


Figure 3: Web forms generation system.

The process begins when the developer(s) interact(s) with an ORM tool to specify a conceptual model. There are several tools supporting ORM [9], including Microsoft's Visio for Entreprise Architects (VEA). The output of this part of the process is a document in an XML-based syntax of ORM, as well as a relational database containing the population of the fact types in the model (if the developer provides such populations). Such XML-based representation of an ORM diagram can be obtained for example by using the Orthogonal Toolbox [13]: an extension of VEA. On the other hand, there are well-known techniques to map ORM diagrams into relational tables [10], and VEA itself implements one of these mappings.

At this point of the process, a developer of a Web page can import the conceptual model into the conceptual model marker in order to specify the structure of a form by specifying presentation markup information. The output of this marking up process is an XML schema containing the marked up portion of the model, and an XML document containing the presentation markup information. This markup information describes how the elements of the

schema are mapped into elements of the form, how the elements of the form are constrained and inter-related, and how the contents of the form elements (e.g. the choices available in the menus present in the form) are extracted from any enumerated value restriction defined in the ORM diagram, as well as from the population of the fact types stored in the database.

The presentation markup information is then given as input to the Forms Generator module, which uses this in conjuncton with the population contained in the database, to produce an XForms description. This XForms code can be included within the body of an XHTML page. Eventually, this XHTML page is served to an end user who fills up the form. The data entered by the user are then submitted to the Forms Processing Servlet, which formats them as an XML document. This XML document is an instance of the XML schema generated by the Model Marker.

A developer can mark up the same model in different ways. The resulting annotated (i.e., marked-up) conceptual models are then processed separately by the Forms Generator, leading to different XForms documents. As an example, we consider the case of a model capturing a set of product categories, where each category is modelled as a type within a type hierarchy: the type Item appearing at the top of the hierarchy, and the most specific product types such as Car, Mobile Phone, or PDA appearing as leaves. Each type inherits the roles of its supertype(s) while having its own roles (e.g. Mobile Phone will inherit the role "has price" from Item while having its own roles such as "has model" and "has network"). Given this model, a forms developer can generate a separate Web form for each of the leaf categories (e.g. one for Mobile Phone, one for PDA, etc.), so that an end user intending to designate a mobile phone will get a different form than another user intending to designate a PDA.

4.2 The conceptual model marker

The model marker is a form-based interface that assists the forms developer throughout the markup process. In other words, the model marker is a form-based interface for generating forms.

The markup process comprises the following 3 major steps, each of which is performed by a separate Web form. The Web forms are displayed one after the other in a wizard-style interaction.

1. The set of entity types available in the schema are displayed in a single-choice menu. The developer selects one of these entity types (say E) as being the main entity type of the form to be generated.

2. The roles played by type E are displayed in any order in a single-choice menu M1. This single-choice menu M1 is connected to another single-choice menu M2 which is initially empty. Two buttons appear between these two menus: one labelled "Add" and the other labelled "Remove". When the "Add" button is pressed, the role name currently selected in the menu M1 is moved to the bottom of the list of role names in M2. When the "Remove" button is pushed, the role name currently selected in M2 is moved to the bottom of the list of role names in M1. A third button "advance" allows the developer to go to the next step. When the "Advance" button is pressed, the list of role names displayed in the menu M2 are taken to be those selected for creating form elements.

3. For each role R selected in the previous step, the following form control elements are displayed:

(a) A required input field where the developer will enter the caption of the form control to be generated for role R.

(b) Two optional text areas corresponding to the description and the documentation of the form control to be generated for role R.

(c) If the entity type that plays the co-role of R has at least one complex identification scheme, a single-choice menu with caption "designation by" and with 2 options: "identifier" and "sub-form".

If the user selects the option "identifier" in the menu of item 3c above, a single-choice menu labeled "Form control element type" is displayed. This menu contains a subset of the following options: "input", "secret", "text area", "upload", "range", "select one", "select many". The subset of the above options that is made available depends on whether there is a uniqueness constraint over R or not, and on the value type associated with the entity type playing the co-role of R (called E' subsequently). Specifically:

- Input, secret, text area, and upload are always available. If there is no uniqueness constraint over R, these options will be annotated with the keyword (repeated), meaning that in the generated form, the chosen form control type will be embedded in a repeat statement (i.e. several input fields will appear instead of only one).
- Range is only available if the value type associated to E' is linearly ordered (e.g. it is a numeric type), and there is a value constraint of type range attached to E' (e.g. it is restricted to the range 1..100).
- SelectOne is only available if the cardinality of R is "one" (i.e. there is a uniqueness constraint attached to R alone), and there is an enumerated value constraint (see [10] page 220) attached to E' .
- SelectMany is only available if the cardinality of R is "many" (i.e. there is no uniqueness constraint attached to R alone), and there is an enumerated value constraint attached to E'.

If on the other hand the user selects "sub-form" in the menu of item 3c above, a new window is opened in which all the above markup process is repeated, starting from step 2 and taking as main entity type, the entity type playing the co-role of R.

4.3 From ORM constraints to XForms constraints

Mandatory role constraints, subset constraints, and exclusion constraints over marked up roles, are translated into XForms "required" and "relevant" constraints over form controls. Specifically, the mandatory role constraints and subset constraints map into "required" constraints, while the exclusion constraints map into "relevant" constraints. For example, a mandatory role constraint over two roles r1 and r2, which are mapped respectively into form control elements f1 and f2, will be expressed in XForms by declaring that f1 is required if f2 has not been filled in, and vice-versa, so that one of these two elements has to be filled in before the submission of the form. Similarly, an exclusion constraint involving r1 and r2 will be translated in XForms by declaring that the form control f1 (corresponding to r1) is irrelevant if the form control f2 (corresponding to r2) has been filled in, and vice-versa. Finally, a range

value constraint (see [10] page 220) associated with an entity type in the conceptual model, is translated into an "IsValid" constraint (see Section 3.1) in the generated form.

The algorithm for mapping ORM constraints into XForms constraints is given below. In this algorithm, the term *field* is used as a synonym of "form control element". The algorithm takes as parameters a set of constraints over roles, and a mapping between these roles and the form controls to be included in the generated Web form. Given these parameters, it computes for each of these form controls, the predicates that will be associated to its *relevant*, *required*, and *IsValid* constraint bindings (see Section 3.1 for a definition of these constraints). The predicates are built using the operators of propositional logic, where the atomic formulae are expressions of the form defined(f), f being an identifier of a field. Such predicates can be readily translated into XPath expressions that can be included in the generated XForms code.

Type definitions */* each type is defined by a set of attributes */*

Type MandatoryRolesConstraint
  roles : Set of RoleID
Type RoleInclusionConstraint
  subset : RoleID
  superset : RoleID
Type RoleExclusionConstraint
  role1 : RoleID
  role2 : RoleID
Type RangeValueConstraint
  role : RoleID */* the value constraint restricts the set of entities that can play the co-role of this role. */*
  ranges : Set of Interval; /* represented by their lower and upper bounds */
Type Predicate = String
/ Predicates are represented as strings using the following syntax:*
  <Predicate> ::= <OR_Predicate>
  <OR_Predicate> ::= <AND_Predicate>
    | <AND_Predicate> 'or' <OR_Predicate>
  <AND_Predicate> ::= <NOT_Predicate>
    | <NOT_Predicate> 'and' <AND_Predicate>
  <NOT_Predicate> ::= <Simple_Predicate> | 'not' <NOT_Predicate>
  <Simple_Predicate> ::= 'defined' '(' <field_identifier> ')'
    | '(' <Predicate> ')' */

Inputs
RoleToField: A function RoleID → FieldID
  / RoleToField is the mapping between role identifiers and field identifiers */*
Roles: Set of RoleID */* the domain of function RoleToField */*
MC : Set of MandatoryRolesConstraint */* the mandatory roles constraints */*
SC : Set of RoleInclusionConstraint */* the role inclusion constraints */*
EC : Set of RoleExclusionConstraint */* the role exclusion constraints */*
Outputs
Required: A function FieldID → Predicate
Relevant: A function FieldID → Predicate
  / These functions provide, for each field, the predicate that will determine whether this field is required/relevant or not */*

Internal variables
mc: MandatoryRolesConstraint
sc: InclusionConstraint
ec: ExclusionConstraint
vc: RangeValueConstraint
r, r': RoleID
conjunction : Predicate
Procedure
for r in Roles
  Required(RoleToField(r)) := "False"
  Relevant(RoleToField(r)) := "True"
/* *The following loop states that for each mc in MC, and for each r in mr.roles:*
$Required(RoleToField(r)) :=$

$$Required(RoleToField(r)) \vee \left(\bigwedge_{r' \in mr.roles, r' \neq r} \neg defined(RoleToField(r')) \right)$$

In other words, a field involved in the mandatory constraint is relevant if none of the other involved fields has been filled in. */
for mc in MC
  for r in mc.roles
    conjunction = "true"
    for r' in mc.roles where r' != r
      conjunction := conjunction + " and " + "not defined(" + RoleToField(r') + ")";
    Required(RoleToField(r)) := Required(RoleToField(r)) + " or " + conjunction
/* *The following loop states that for each sc* $\in$ *SC:*
$Required(RoleToField(sc.superset)) :=$
$Required(RoleToField(sc.superset)) \vee defined(RoleToField(sc.subset))$
In other words, when the field corresponding to the subset role is filled in, the field corresponding to the superset role has to be filled in as well. */
for sc in SC
  Required(RoleToField(sc.superset)) := Required(RoleToField(sc.superset)) +
                                        " or defined(" + RoleToField(sc.subset) + ")"
/* *The following loop states that for each ec* $\in$ *EC:*
$Relevant(RoleToField(ec.role1)) :=$
$Relevant(RoleToField(ec.role1)) \wedge \neg defined(RoleToField(ec.role2))$
and conversely that:
$Relevant(RoleToField(ec.role2)) :=$
$Relevant(RoleToField(ec.role2)) \wedge \neg defined(RoleToField(ec.role1))$
So that when the first field is filled in, the other becomes irrelevant and vice-versa. This can be generalised to exclusion constraints involving more than two roles. */
for ec in EC
  Relevant(RoleToField(ec.role1)) := Relevant(RoleToField(ec.role1)) +
                                  " and not defined(" + RoleToField(sc.field2) + ")"
  Relevant(RoleToField(ec.role2)) := Relevant(RoleToField(ec.role2))+
                                  " and not defined(" + RoleToField(ec.role1) + ")"

```
/* The following loop states that for each vc ∈ VC:
   IsValid(RoleToField(vc.role)) := ⋁_{tp∈vc.ranges} tp.inf ≤ RoleToField(vc.role) ≤ tp.sup
*/
for vc in VC
  IsValid(RoleToField(vc.role)) := "false" /* initialisation */
  for tp in vc.ranges
    IsValid(RoleToField(vc.role)) := IsValid(RoleToField(vc.role)) + " or " +
                                     RoleToField(vc.role) + "≥" + tp.inf + " and " +
                                     RoleToField(vc.role) + "≤" + tp.sup
```

Enumerated value constraints (such as those appearing under the types Make, Model, Finish and BodyStyle in Figure 1) are taken into account either by including the enumerated values as options within "SelectOne" or "SelectMany" form control elements, or by translating them into "IsValid" constraints, in a similar way as range value constraints are translated by the above algorithm.

Finally, equality constraints are first translated into pairs of subset constraints, which are then translated into "Required" constraints over fields by the algorithm.

5 Related work

To the best of our knowledge, the issue of generating Web forms from conceptual models has not been addressed in a comprehensive way by previous works. A notable exception is [3], in which the author proposes a methodology for modeling Web pages using UML. In particular, the author introduces a collection of UML stereotypes and icons for modelling Web forms. These include a <<Form>> stereotype to declare that a class models a form, a <<Submit>> stereotype to declare that a given method will process the submission of a form, and other presentation stereotypes such as <<Text>>, <<Checkbox>>, etc. This approach differs from ours in that the form control elements are explicitely specified in the conceptual model, whereas in our approach, the conceptual model makes no reference to the form controls to which the entity and fact types are mapped: this information is specified after the design phase through a markup process, thereby leading to more reusable conceptual models. Also, our approach has the advantage that it translates the constraints expressed in the conceptual model into constraints in the Web form, whereas [3] does not.

Although in this paper we focused on the generation of XForms code, it is possible to apply the same ideas to generate Web forms in other languages such as HTML/Javascript (as sketched in [8]) or UIML [11]. More generally, the generation of Web forms from conceptual models is a particular case of a more general problem: that of mapping conceptual models into user interface models. Visual browsers, visual editors and visual query languages (e.g., SUPER [7]) implictely implement such mappings: they automatically create a user interface for database insertion, deletion, update, and retrieval, from a given conceptual data model. In the specific context of ORM, [2] describes a procedure for grouping the types of a conceptual model, in a way that facilitates the development of application forms for database manipulation. In particular, this procedure includes automated steps for: (i) anchoring fact types to entity types so that a fact type appears within the form of the entity type to which it is anchored; (ii) determining the order in which database insertions should be performed; and (iii) detecting situations where deletions are to be cascaded. The "external" model generated

through this procedure is then linked to the "internal" (i.e. database) model generated through an ORM to relational mapping, so as to maintain the consistency between these models (e.g. when an object is deleted in one model, it is also deleted in the other model).

None of these previous works however, addresses the issue of encoding constraints expressed in a conceptual model, in a language for forms description such as XForms.

The issue of annotating XML schemas with presentation information, database mappings, and other types of external information, is the subject of ongoing efforts. The Schema Adjunct Framework (SAF) [15] is an XML-based language used to associate domain-specific data with XML schemas and their instances. In a nutshell, a schema adjunct is a set of of data items associated with the elements of an XML schema, that describe a particular kind of processing. Schema adjuncts can be used for example to describe a mapping of an XML schema to a relational database, or a transformation method from an XML schema to a Web form. In particular, the presentation markups required to generate Web forms in the architecture of Figure 3 could be represented in a schema adjunct, so that a conceptual schema can be captured in a non-interactive way. The SAF API can then be used to process these markups.

Finally, the issue of generating Web forms from ORM has some connections with that of generating database schema definitions or XML schema definitions from ORM. In this context, the work by Bird et al. [1] on generating XML schemas from ORM diagrams has some commonalities with ours, to the extent that the underlying idea is to translate constraints encoded in ORM into constraints in an implementation-level language for Web-based systems. In the future, we can expect that several other mappings from ORM to implementation languages for Web systems (e.g, RELAX NG [12]) will be developed.

6 Conclusion

The main contribution of this paper is a comprehensive approach to markup (i.e. annotate) conceptual models in order to generate Web forms in a constraint-preserving manner. By constraint-preserving we mean that the constraints expressed in the conceptual schema are encoded as structural and active features within the form, in such a way that the data entered by the user can be interactively validated against the constraints. We specifically studied a mapping from ORM (as a conceptual modelling language) to XForms (as a Web form description language). However, the underlying principles apply to other source languages (e.g. UML) and target languages (e.g. UIML).

A direction for future work could be to apply a similar approach in order to define transformations from conceptual models to other (Semantic) Web languages, including schema languages such as RELAX NG, and extensions of RDFS (e.g. DAML+OIL).

Another direction for further work could be to explore the application of the proposed markup approach to the generation of multi-lingual Web interfaces from conceptual models and ontologies. The general idea here is to integrate conceptual modelling techniques with RDF and Semantic Web technologies (e.g. taxonomies, lexical databases, thesaurus, etc.) in order to deliver high-level user interfaces to heterogeneous Web information sources.

Acknowledgment This work was supported by the Australian Research Council SPIRT Grant "Self-describing transactions operating in a large, open, heterogeneous, distributed environment" involving QUT, UNSW, and GBST Holdings Pty Ltd. The authors wish to thank Mitra Heravizadeh for her participation in the early stages of this work.

References

[1] L. Bird, A. Goodchild, and T. Halpin. Object Role Modelling and XML-Schema. In *Proceedings of the 19th International Conference on Conceptual Modelling (ER)*, Salt Lake City UT, USA, October 2000. Springer Verlag.

[2] L. Campbell and T. Halpin. Automated support for conceptual to external mapping. In *Proceedings of the 4th Workshop on Next Generation CASE Tools (In conjunction with CAiSE'93)*, Paris, France, 1993.

[3] J. Conallen. Modeling Web applications architecture with UML. *Communications of the ACM*, 42(10), October 1999.

[4] D. Connolly, F. van Harmelen, I. Horrocks, D.L. McGuinness, P.F. Patel-Schneider, and L.A. Stein. DAML+OIL (March 2001) Reference Description. Note, W3C Consortium, December 2001. Accessed from `http://www.w3.org/TR/daml+oil-reference` on 11 March 2002.

[5] S. Cranefield, S. Haustein, and M. Purvis. UML-based ontology modelling for software agents. In *Proceedings of the Workshop on Ontologies in Agent Systems (in conjunction with AGENTS'2000)*, Montreal, Canada, May 2001. Accessed from `http://autonomousagents.org/2001/oas` on 25/06/2002.

[6] J. Demey, M. Jarrar, and R. Meersman. A Markup Language for ORM Business Rules. In *International Workshop on Rule Markup Languages for Business Rules on the Semantic Web*, Sardinia, Italy, May 2002. Accessed from from `http://www.soi.city.ac.uk/~msch/conf/ruleml` on 25/06/2002.

[7] Y. Dennebouy, M. Andersson, A. Auddino, Y. Dupont, E. Fontana, M. Gentile, and S. Spaccapietra. SUPER: visual interfaces for object + relationship data models. *Journal of visual languages and computing*, 6(1):27 – 52, 1995.

[8] M. Dumas, L. Aldred, M. Heravizadeh, and A. ter Hofstede. Ontology markup for web forms generation. In *WWW'2002 Workshop on RDF and Semantic Web Applications*, Honolulu HI, USA, May 2002.

[9] T. Halpin. ORM: Object Role Modeling. `www.orm.net`.

[10] T. Halpin. *Information Modeling and Relational Databases*. Morgan Kaufmann, 2001.

[11] M. Abrams and J. Helms (Eds.). UIML 3.0 Language Specification, February 2002. Accessed from `http://www.uiml.org` on 28 June 2002.

[12] OASIS. RELAX NG Specification. Dowloaded from `http://www.oasis-open.org/committees/relax-ng` on 3 July 2002.

[13] Orthogonal Software Co. The Orthogonal Toolbox. Accessed from `http://www.orthogonalsoftware.com` on 1 July 2002.

[14] J. Rumbaugh, I. Jacobson, and G. Booch. *The Unified Modeling Language Reference Manual*. Addison-Wesley, 1999.

[15] S. Vorthmann, J. Robie, and L. Buck. The Schema Adjunct Framework. Note, W3C Consortium, July 2001. Accessed from `http://www.w3.org/TR/daml+oil-reference` on 12 March 2002.

[16] W3C Consortium. XForms — The next generation of Web forms, 1999-2002. `http://www.w3.org/MarkUp/Forms`.

Real World Semantic Web Applications
V. Kashyap and L. Shklar (Eds.)
IOS Press, 2002

The "Emergent" Semantic Web: A Consensus Approach for Deriving Semantic Knowledge on the Web

Clifford BEHRENS
Telcordia Technologies, Inc
445 South Street
Morristown, NJ 07960-6438, USA
cliff@research.telcordia.com

Vipul KASHYAP
National Library of Medicine
8600 Rockville Pike
Bethesda, MD 20894, USA
kashyap@nlm.nih.gov

Abstract. The recent and growing interest in the Semantic Web has given rise to a flurry of activity in standardization bodies (such as the W3C) to specify semantics using formal languages and inference mechanisms. The real challenge, however, is to link formal semantics with deeper meaning as reflected by consensus discovered among users on the Semantic Web. We believe the process of deriving and formally describing ontologies for the Web (expressed using standardized languages) is necessarily a social-cultural one; hence, new consensus-based tools are required to derive shared semantic systems for different communities of interest. This paper introduces Consensus Analysis as a means for deriving semantic knowledge from the information provided by subject matter experts and describes the *Schemer System* prototype for acquiring and processing this information. The results of a trial application of this approach and prototype on technologists asked to identify current mass market consumer trends in the domain of Internet privacy and security are reported. These findings implicate Consensus Analysis as a powerful tool capable of enabling the semantic Web by yielding core knowledge such as controlled vocabularies and domain ontologies.

Introduction

Recently there has been a great interest in the Semantic Web and issues related to specification and exploitation of semantics on the WWW. In particular, shared ontologies are being proposed for representing the core knowledge that forms the foundation for semantic information on the Web. Fensel [1] has identified two broad research thrusts related to ontologies:

1) Approaches to standardize the formal semantics of information to enable machine processing. Work being done as a part of the W3C RDF working group [2] and the DAML+OIL initiative [3] falls within this category.
2) Approaches to define real world semantics linking machine processable content with meaning for humans based on consensual terminologies.

To realize the goals of the Semantic Web, there is a need to wed approaches centered on the formal representation of semantics with approaches to systematically acquire terminologies that best express shared systems of meaning among users. We believe the process of deriving and describing domain ontologies necessarily involves the search for consensus among domain experts and, therefore, is inherently a social-cultural one. As such, the proper approach to deriving knowledge, like domain ontologies, ought to be sensitive to the semantic context of information, and should be informed by the real-world bottom-up, decentralized process in which knowledge typically evolves. After all, decentralized models for consensus achievement better reflect the dynamic sociological characteristics of the Web (which have been the cause for its rapid acceptance and success). In this manner more meaningful ontologies (expressed in standardized formal languages) can emerge through more natural interactions of Web users within their respective communities. We agree with Fensel [1] who claims that the real challenge for making the semantic Web a reality is, "a model for driving the network that maintains the process of evolving ontologies."

Consistent with this claim, similar interest in Knowledge Management processes has motivated new research in automatic knowledge acquisition, classification, and representation [4]. Much of the discussion on Knowledge Management has focused on information technology, e.g., hardware, software and communications networks, but has not laid out in clear terms what notions of "knowledge" need to be supported by this technology. For example, there exists in the literature a recurring theme that knowledge is any information stored in a Knowledge Repository, and that this knowledge can somehow be acquired or "discovered" automatically from disparate, heterogeneous information sources, e.g., Web pages and networked document collections. This approach seems at best naïve as it ignores the context and intended purpose of source information. Without establishing this context and purpose, it seems unlikely that much useful "knowledge" can be discovered as it leaves matters pertaining to information's meaning and relationships with existing knowledge open to broad interpretation.

Within the literature there is expressed the idea that not all information is knowledge; information only becomes knowledge once it is mapped to a knowledge structure, i.e., it is organized in a way that makes it accessible and comprehensible to users [4]. In fact, this qualification suggests that there may exist many such structures for organizing the same information, again supporting the idea that context and purpose are essential for transforming information to knowledge. It also implicates the importance of knowing the "community of interest" (COI) for both the producer and consumer of information to enable this transformation since members of different COIs may set different contexts or use the information with very different intentions. In the emergent Semantic Web, it is critical to determine the "consensus" knowledge structures for a COI.

The term "community" is becoming ubiquitous, particularly in discussions related to delivery of personalized services on the Internet, yet there exist distinct usages of the term. For some, a community seems to consist of all who, because of a shared interest in certain kinds of information, frequent the same place, real or virtual, regardless of any interaction among them, e.g., all those who browse the same Web site [5]. A more social usage entails information exchanges among a collection of individuals, e.g., all those who exchange useful information about some topic of mutual interest through email or chat rooms [6]. A more sophisticated "cultural" notion of community refers to all who, in addition to meeting the two preceding conditions, share a vocabulary, *semantics* and theory for organizing information.

For members of this class of community there exists some common purpose and key concepts for communicating ideas and sharing experiences. In this paper, this last notion of community is adopted because, as it will be demonstrated, it provides an opportunity to analyze knowledge, and its variations among individuals, with greater formal rigor. It also helps to more clearly draw the lines operationally between information, individual knowledge, and what will be referred to later as "cultural knowledge."

Much research has already been conducted in the social sciences, particularly cognitive anthropology and cognitive psychology, on modeling knowledge domains, i.e., conceptual categories that include other semantically-related categories, and eliciting the information needed to build these models. However, most of these methods are extremely time-consuming, taxing the attention of a few SMEs (Subject Matter Experts), those recognized as experienced and possessing specialized domain knowledge. *Consensus building* is another approach to building knowledge representations that is gaining increasing popularity in the Information Processing standards community and elsewhere [7, 8]. New information technology could be applied to eliminate much of the need (and enormous cost) of face-to-face group decision-making meetings, e.g., read [9] and [10] for examples of IT approaches to collaborative knowledge construction.

Previous methods for building knowledge from consensus have been tried, e.g., Delphi approaches [11, 12], but these are typically iterative and require much human intervention. While the importance of consensus to achieving views that best represent collective thinking is often stressed, too often views are biased strongly by the force of individual personalities and are not representative of any particular COI. Other problems arise from the heterogeneous composition of decision-making groups whose members conceptualize the same problem from widely different perspectives, i.e., those of different COIs. Moreover, simple polling methods that only average expert opinion do not usually yield results with the depth and logical properties of real domain knowledge, nor do they exploit the contributions of the most competent SMEs. Thus, there is need for a different approach that derives, rather than forces, consensus views, does so without the need for much human intervention and many iterations, acquires useful information from SMEs (weighted by their competence) at their convenience, and is capable of yielding shared knowledge for a demonstrable domain of interest.

By combining new formal and more rigorous approaches to consensus-modeling, specifically powerful methods of Consensus Analysis that already have been tested successfully by Cognitive Anthropologists in numerous knowledge domains, the network services approach taken in this research overcomes the limitations of previous computer-assisted approaches. This is accomplished by (1) incrementally refining or "evolving" knowledge, (2) providing *metrics* for evaluating the cultural saliency of a domain and the knowledge-based competency of SMEs in a COI, (3) dynamically assigning SMEs to the most "appropriate" COI and (4) not only spreading the task of knowledge acquisition among many SMEs, rather than just a few, but also leveraging Web infrastructure to engage them at their convenience.

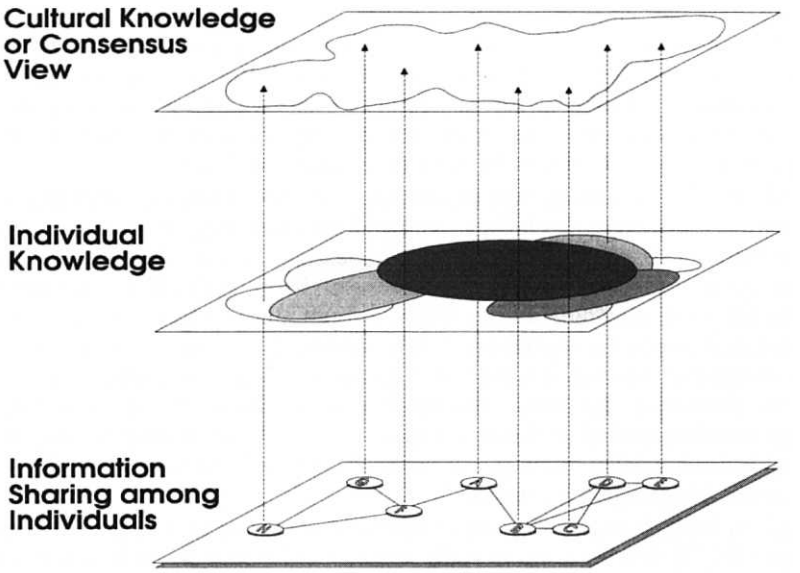


Figure 1. Knowledge distribution, knowledge sharing and consensus.

1. Cultural Knowledge and Consensus Analysis

Consensus Analysis is based on a few simple, but powerful, ideas , i.e., *knowledge is both distributed and shared* [13]. For any knowledge domain, and any group of subjects "expert" in this domain, so-called "SMEs" possess different experiences; hence, they know different things, and some of them know more than others (see Fig. 1). Information sharing, e.g., among individuals A-H in the figure, facilitates the availability of a much larger pool of information with non-uniform distribution of knowledge across SMEs. For example, many information standards groups are composed of data providers, data users, librarians and software vendors. These groups tend to possess different experiences with data, and apply their own unique views and semantics to describe these data. Yet, certain individuals (the hi-tech "gurus") are recognized as being more knowledgeable than others, i.e., there exist recognized domain "experts." Because of their widely regarded and highly-valued knowledge, these experts are frequently requested to share what they know with others as consultants or as leaders in standards-setting groups, or render opinions about how best to describe or classify information in their domain of expertise. Hence, one typically finds within any COI that there is differential expertise among its members, but also some knowledge that is widely-shared and recognized as being "essential." In fact, this knowledge may be so fundamental and its use so widespread that, over time, it becomes logically well-structured or canonical, e.g., even published as a metadata content standard. *The process of mapping information onto such a consensus standard is the essence of cultural knowledge creation.*

Cultural knowledge is not all that one knows (e.g., the set of knowledge for each individual represented in the middle layer of Figure 1); nor is it the sum total of what everybody knows (e.g., the union of individual knowledge sets in the middle layer). Rather, it is an abstraction, knowledge shared in its "broad design and deeper principles" by members of a society or community [14]. In other words, while its entire details are not usually known (or can be always be articulated explicitly) by anyone, cultural knowledge consists of those things that all members of a COI understand all others hold to be true.

Kroeber [15] referred to this highly-structured, rich form of knowledge as a "systemic culture pattern," a coherent subsystem of knowledge that tends to persist as a unit. This unit features a semantic system, consisting of an appropriate vocabulary and grammar, for classifying and talking about elements within a knowledge domain. Examples of cultural knowledge are: a kinship terminology [16, 17], or perhaps a metadata content standard [18], a consensus statement for screening cancer [19], or a set of software requirements [20]. It is this shared pool of structured information, acquired primarily by learning, which constitutes cultural knowledge [21].

1.1 Robustness of the Consensus Model

The significance of information sharing and distribution of cultural knowledge has encouraged some researchers to exploit consensus, measured by intersubject agreement, as an indicator of knowledge. The method of Consensus Analysis was first presented in several seminal papers [13, 22, 23]. In addition to introducing the formal foundation for Consensus Analysis (reviewed later in more detail), the initial papers cited above also provided examples of its application to modeling knowledge of general information among US college students, and the classification of illness concepts among urban Guatemalans. Other more recent applications of Consensus Analysis have focussed on measuring cultural diversity within organizations [24]. These successes, obtained for a wide variety of domains and social-cultural contexts, indicate that the following three explicit assumptions, upon which Consensus Analysis is based, are extremely robust [13]:

i) *Common Truth.* There is core knowledge (expressed in a highly probable set of answers to questions or "items") that is "applicable" to all SMEs or, put another way, all SMEs are members of the same COI and generally share a common perspective or "cultural reality."

ii) *Local Independence.* The information or responses provided by each SME are acquired independently from those of other SMEs, i.e., SME item response random variables satisfy conditional independence for all possible response profiles and the core answer set.

iii) *Homogeneity of Items.* Each SME has a fixed level of "competence" or "expertise" across all items, i.e., items used to sample what SME's know are equally difficult and provide representative coverage of a coherent domain. In practice, this assumption has been found to be quite robust and requires only that those SMEs who are most knowledgeable in a domain consistently outperform non-experts.

From these assumptions, it is possible to derive a method for estimating three properties of interest: (1) a measure of the overall saliency of the knowledge domain represented by the pool of items, (2) the level of domain expertise or "cultural competence"

for each SME based on the amount of consensus or agreement between his/her responses to items with those of all other SMEs, and (3) the most probable set of "correct answers," inferred from the responses of each SME and weighted by their respective competence measures, i.e., the consensus view.

1.2 Statistical Methodology

As mentioned earlier, the Consensus Analysis Model can be derived formally from the three assumptions given in section 2.1. This formal model consists of a data matrix X containing the responses X_{ik} of SMEs *1..i..N* on items *1..k..M*. From this matrix another matrix M^* is estimated and it holds the empirical point estimates M^*_{ij}, the proportion of matching responses on all items between SMEs i and j, *corrected for guessing* (if appropriate), on off-diagonal elements (with $M^*_{ij} = M^*_{ji}$ for all pairs of SMEs i and j). Alternatively, another matrix C^*, which contains the observed covariances C^*_{ij} between the responses of SMEs i and j, corrected for variance among SME answers, may be substituted for M^* [25]. To obtain D^*_i, an estimate of the proportion of answers SME i "actually" knows and the main diagonal entries of M^* (or C^*), a solution to the following system of equations is sought:

$$M^* = D^* D^{*\prime} \text{ or alternatively,} \quad (1)$$

$$C^* = D^* D^{*\prime} \quad (2)$$

where D^* is a column vector containing estimates of individual competencies $D_1..D_i..D_N$ and $D^{*\prime}$ is merely its transpose. Since equation 1 (or 2) represents an overspecified set of equations and because of sampling variability, an exact solution is unlikely. However, an approximate solution yielding estimates of the individual SME competencies (the D^*_i) can be obtained by applying *Minimum Residual Factor Analysis* [26], a least squares approach, to fit equation 1 (or 2) and solve for the main diagonal values. The relative magnitude of eigenvalues (the first eigenvalue λ_1 at least three times greater than the second) is used to determine whether a single factor solution was extracted. All values of the first eigenvector, v_1, should also range between 0 and 1. These results test the validity of the Common Truth assumption.

If the criteria above are satisfied, then the individual SME competencies can be estimated with:

$$D^*_i = v_{1i} \sqrt{\{\lambda_1\}} \quad (3)$$

The D^*_i, then, are the loadings for all SMEs on the first factor. These estimates are required to complete the analysis, i.e., to infer the "best" answers to the items. The estimated competency values (D^*_i) and the profile of responses for item k ($X_{ik,l}$) are used to compute the Bayesian *a posteriori* probabilities for each possible answer. The formula for the probability that an answer is "correct" follows:

$$Pr(<X_{ik}> i=l \mid Z_k=l) = \prod_{i=1}^{N} [D^*_i + (1-D^*_i)/L]^{X_{ik,l}} [(1-D^*_i)(L-1)/L]^{1-X_{ik,l}} \quad (4)$$

where Z_k is the "correct" answer to item k, l is the l^{th} response to item k, and L is the total number of possible responses $(l_1...l_L)$ to item k. Again, it should be mentioned that the "correctness" of an answer is relative to the perspective shared by members of a particular COI, i.e., the one sampled. Equations 1-4 provide formal motivation for the approach taken in this research, and indicate algorithms that need to be implemented in software as part of a network-enabled consensus server.

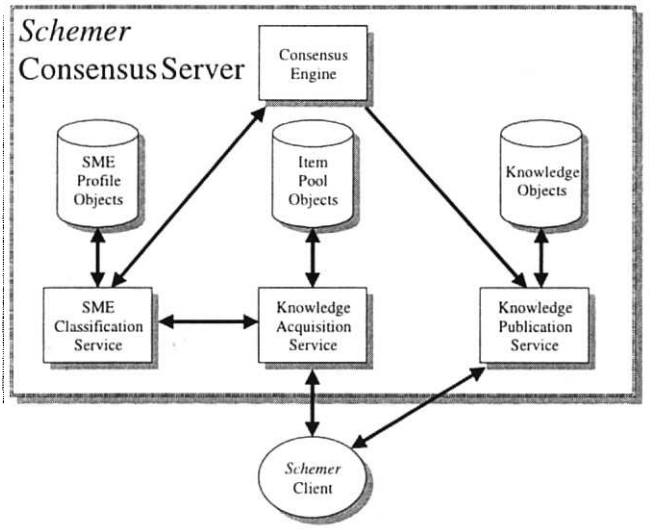


Figure 2. *Schemer* system architecture.

2. System Architecture and Prototype

Telcordia researchers have begun to design a software prototype called the *Schemer System*, shown in Fig. 2. Key software components in this design have already been implemented to communicate better some of the objectives of the approach, stimulate greater interest in it, and demonstrate the feasibility of automating Knowledge Acquisition and Consensus Analysis modeling. Future work will include development of Publication Services and a fuller integration of software components in a continuous Web-based service.

In our current design, the *Schemer System* consists of a *Schemer Client* and *Schemer Server*; however the latter really involves the interaction of four services: a *Subject Matter Expert Classification Service*, a *Knowledge Acquisition Service*, a *Consensus Engine*, and a *Knowledge Publication Service*. These services read and write information to several data bases, one storing information about SMEs, another storing pools of items used to acquire information from SMEs, and another which stores the derived knowledge structures, i.e., the controlled vocabularies, forecasts, ontologies, classification schemes, or productions systems. Next, each of these services is examined in more detail.

2.1 Client

The job of the *Schemer Client* is to provide a graphical/text interface through which a user communicates with the *Schemer Server*. It presents information sent by the Server such as item forms and knowledge visualizations, both textual and graphic.

2.2 SME Classification Service

This service determines a knowledge domain of interest for a SME, and assigns a SME to his/her proper COI. Classification is necessary to present a SME with meaningful items and knowledge derived from a consensus analysis of peer responses. Knowledge domain identification may be determined either by asking a SME to select a known knowledge domain from a list or, if unknown to *Schemer*, the SME is asked to input the name of this knowledge domain. COI classification may be accomplished in two stages, *a priori* and *post hoc*. If nothing is known about the SME, then preliminary COI classification is made by asking the SME to choose a COI from a list of COIs already known to *Schemer*. If, however, the SME cannot find an appropriate COI in this list (or if the knowledge domain is unknown to *Schemer*), then he/she is prompted for a list of key terms frequently used to describe objects in the knowledge domain of interest. These terms are matched against key term lists (if they exist) for known COIs to determine the "best" COI classification for this SME. But once an item form has been constructed for this SME and used to acquire more information, *post hoc* analysis of results obtained from Consensus Analysis may be used to reclassify this SME, if he/she so chooses. To compare new information obtained from the SME with information known for SMEs already classified, the *Subject Matter Expert Classification Service* reads the data it needs from a repository of SME profile objects.

2.3 Knowledge Acquisition Service

Based on the knowledge domain and preliminary COI classification obtained for a SME, this service selects an appropriate item pool object and composes an instrument or form that is used to determine what the SME knows about the domain. Items published on these forms are read from a repository of item pool objects, each identified by knowledge domain and

COI. This service sends the form to the Client where the SME enters his/her responses to items on the form, then sends these back to the *Knowledge Acquisition Service*. The SME's response pattern, along with his/her ID, is stored in a repository of SME profile objects, also grouped by COI and knowledge domain.

2.4 Consensus Engine

This service performs a Consensus Analysis of data collected for a knowledge domain/COI grouping each time new responses are added to a SME's profile, and stores the updated result in a *Knowledge Repository*, along with ancillary statistics, e.g., Goodness-of-fit indices. Not only does the *Consensus Engine* analyze data read from SME profiles, but it also adds information to these, e.g., a SME's competency score.

2.5 Knowledge Publication Service

On a user's request, this service constructs forms with textual and graphical representations of derived knowledge, stored in the *Knowledge Repository*, for presentation on the *Schemer Client*. Access to information stored in this repository is also provided by this service so that a user can retrieve a knowledge object for use with his/her own software application.

To date, a skeleton *Knowledge Acquisition Service* has been built, capable of taking as input from a SME's Web browser a knowledge domain and COI value, then return a form with an appropriate item set for this knowledge domain/COI combination. Currently, only dichotomous (True/False) formats are supported. Once a SME completes this form and submits his/her responses to the *Knowledge Acquisition Service*, it notifies the *Consensus Engine* that a SME's profile has been updated. The *Consensus Engine* processes all of the response vectors for SMEs in the same knowledge domain/COI data base, then stores the results, e.g., eigenvalues, SME competency scores and the estimated answer key, in the knowledge base. All of these services have been implemented in Java® and the R® statistical programming environment, so can run under Unix® or Windows®.

3. Experiment

The remainder of this paper describes an experiment that was conducted among Telcordia technologists to derive a consensus view of mass-market consumer trends related to Internet security and privacy. While no attempt will be made to derive a domain ontology from this experiment, our intent is to demonstrate how Consensus Analysis works and to further suggest that it seems well-suited for this purpose.

A prototype of the *Schemer* system was built and used to deliver a questionnaire consisting of sixty-seven items related to privacy and security of information on the Internet (see Appendix A). These items were derived from Georgia Tech's *10th World Wide Web User Survey*, which includes a section entitled, "Online Privacy and Security". A request

was mailed electronically to Research Scientists belonging to two labs within Telcordia Technologies Applied Research. These sample SMEs were asked to answer items on the questionnaire *as if they were domain experts* being asked for their opinions about *mass market consumer trends within the Web user community*, not necessarily with their personal opinion. Along with responses to the questionnaire, SMEs were asked for their employee ID and a list of no more than twenty descriptors that they believed best represented their professional area of expertise. The former was used as a pointer to other ancillary information about the SME, e.g., lab, department, group, and office location, while the latter was collected to help associate the domain expertise of a SME with that of others in the sample. A total of thirty-six Research Scientists responded to the request above. This sample was opportunistic, not random; moreover, a special request was made to members of Telcordia's *Computer Networking Research* department, which specializes in Internet security issues, so that the responses of these domain experts could be compared to others in the sample.

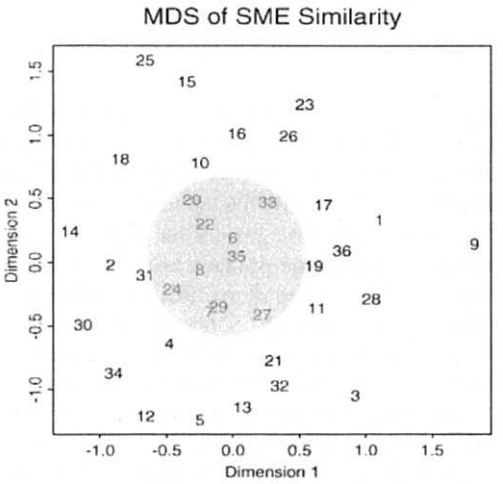


Figure 3. Plot of MDS results showing similarities in responses among SMEs. Similar SMEs are plotted close to one another. Stress= 0.260 after 19 iterations.

The similarity or agreement among SME response patterns can be explored in Figure 3. This two-dimensional plot was obtained from a Multidimensional Scaling of only the off-diagonal entries of the consensus matrix (M_{ij}* in Equation 1) calculated for the thirty-six Telcordia SMEs [27]. In this visualization, the SMEs with similar responses are plotted closest to one another. The absence of clustering in this plot suggests that the study sample was drawn from a single COI whose members share core domain knowledge about "Online Privacy and Security." This notion was tested more rigorously by estimating a consensus model for these data.

3.1 Knowledge Domain Validation

As the review of Consensus Analysis pointed out, knowledge derivation rests on establishing the validity of the domain to those SMEs in the sample. This is accomplished by inspecting the relative magnitudes of the eigenvalues for the first factors extracted from the consensus matrix using Minimal Residuals Factor Analysis. Again, the "rule-of-thumb" is that the eigenvalue of the first factor must be at least three times greater than the second; moreover, subsequent eigenvalues should all be small and roughly equivalent. Inspection of the eigenvalues for the first three factors extracted from the response set collected from Telcordia SMEs reveals that the first is over six times greater than the second, and the second and third eigenvalues are almost equal (see Table 1). This lends strong support to the claim that the items on the questionnaire are sampling a single, coherent knowledge domain, and that this domain has salience for the sample of respondents. Moreover, the high Pseudo-Reliability Coefficient (0.944) also obtained suggests that these results are stable and would likely be the same ones obtained with repeated sampling [13].

Table 1. Eigenvalues for testing saliency of "Online Privacy and Security" knowledge domain.

Factor	Eigenvalue	Percent	Cumulative %	Ratio
1	11.902	77.8	77.8	6.500
2	1.831	12.0	89.8	1.175
3	1.559	10.2	100.0	
	15.292	100.0		

3.2 Estimation of SME Competence

Having established the saliency of "Online Privacy and Security" as a knowledge domain for SMEs in the sample, it is possible to estimate each one's competency in this domain. The competencies for this sample of SMEs are listed in Table 2. This metric can be interpreted as the probability that a SME would correctly answer an item. Competencies for this sample range from 0.32-0.76 with a mean of 0.56± 0.11. With a sample size of thirty-six, and average competency level of 0.56, it ought to be possible to correctly classify (as either "true" or "false") at least 95% of the items on the "Online Privacy and Security" questionnaire with a 0.999 confidence level [13].

Table 2. Estimates of competency for thirty-six SME's questioned about "Online Privacy and Security."

SME	Competency	Organization	Location
1	0.48	C2E	M3B
2	0.60	C2E	M3B
3	0.41	C2E	M3B
4	0.56	C2E	M3R
5	0.48	C8E	M3B
6	0.75	ICI	N3X
7	0.75	Missing	Missing
8	0.67	I9B	M2R
9	0.32	C7H	M3B
10	0.58	C8I	N1X
11	0.61	C1B	M3B
12	0.47	C2I	M3B
13	0.52	I0B	M2R
14	0.50	C2I	M3B
15	0.42	C2F	M3R
16	0.52	C8E	M3R
17	0.59	I5I	M2B
18	0.45	C7E	M3B
19	0.59	C8E	M3B
20	0.67	C2I	M3R
21	0.55	C2A	M3B
22	0.67	I5I	M2R
23	0.46	C8F	M3B
24	0.76	C8B	N3Z
25	0.35	I5B	M3R
26	0.51	C8F	M3B
27	0.67	C8F	M3B
28	0.52	I5H	M2R
29	0.72	C7H	M3B
30	0.52	A4B	M2R
31	0.63	Missing	Missing
32	0.58	I9D	M2B
33	0.64	C2A	M3R
34	0.51	I9I	M2B
35	0.69	C2A	M3B
36	0.59	C2A	M3B

3.3 Knowledge Derivation

By using SME competencies as weights, the most probable set of answers can be estimated from SME responses with Bayes' formulation in Equation 4. In Table 3 the answers obtained

Table 3. Cross tabulation comparing majority responses on GVU survey to those estimated from responses of Telcordia SMEs with Consensus Analysis.

GVU Survey	Telcordia SMEs		Marginal Totals
	False	True	
False	20	14	**34**
True	5	28	**33**
Marginal Totals	**25**	**42**	**67**

in this way are compared to the dominant responses given by the 1,482 respondents who completed the GVU survey. Pearson's Chi-square (with Yate's continuity correction) was calculated to test for independence between the two sets of answers [28]. A Chi-square value of 11.852, with one degree of freedom, was obtained from the test, and with a p-value<0.001, there is strong support to conclude that the answers estimated through Consensus Analysis are not different from those obtained for the GVU sample. A Yule's Q= 0.78 also indicates that this association is a reasonably strong one.

3.4 SME Classification

With estimates of SME competencies in hand, the spatial arrangement of points plotted in Figure 3 can be given a particularly nice intuitive interpretation. Those SMEs who knew the most about "Online Privacy and Security" are plotted in the center of this figure; in fact, those ten SMEs with the highest competency scores fall within the shaded area; while those with the lowest scores are located at the periphery of this plot. However, there also exists idiosyncratic variation among these SMEs in what they know about this domain, and so domain expertise seems to cross organizational boundaries. This idea was tested in several ways.

The terms that SMEs provided to describe their technical areas of expertise were carefully enumerated. Surprisingly, while the frequent use of "hot buzz words" was anticipated, it seems that SMEs exploited free-listing as an opportunity to create very specialized identities. In fact this sample of SMEs applied 189 unique descriptors (each consisting of one or more terms) to characterize their expertise. The number of descriptors listed by SMEs ranged from 0-16 with an average list size of 5.25 descriptors. Four SMEs listed no terms. Only nineteen of the 189 descriptors were listed by more than one SME and all but two of these nineteen were listed only twice, further suggesting a reason for the absence of any discernable clustering of points in Figure 3. However, the five most competent SMEs (24, 6, 7, 29, and 35) identified themselves as knowing more about business and marketing aspects of telecommunications, e.g., "Business planning", "economics", "market-oriented programming"; and used terms such as "system administration" and operations "hand-offs", implying greater familiarity with consumer or user-oriented perspectives. The only shared concepts expressed in the free lists of those SMEs (9, 25, 3,

15, and 18) with the lowest competency scores were "distributed computing", "Internet", "Internet Protocols", and to some degree more abstract interests, e.g., "mathematics", "formal methods", and "theory of distributed systems". It seems that this group is focused more on privacy and security from a network engineering or design perspective, rather than from a consumer's point-of-view.

More rigorous statistical tests of the organizational and locational basis for knowledge distribution among these SMEs were also made. For these tests, two other symmetrical distance matrices were constructed: the first from SME organization numbers and the second from their office locations (both listed in Table 2). The Organization Code consists of three characters that identify a SME's lab, department and group, respectively. A matrix, whose cells express the organizational distance between SMEs, was constructed from this information in the following manner: a "0" was assigned to all cells along the superdiagonal, a "1" was entered into a cell for SMEs belonging to the same lab, department and group, a "2" for SMEs belonging only to the same lab and department, a "3" to those SME's belonging only to the same lab, and a "4" to those SMEs in different labs. The Office Address also consists of three characters identifying a SME's office site (two possible sites separated by about 50 miles), floor and wing. A locational distance matrix for SMEs was calculated in the following manner: a "0" was assigned to all entries along the superdiagonal, a "1" was entered in a cell for two SMEs located at the same site, on the same floor, and in the same office wing, a "2" for SMEs occupying only the same site and floor, a "3" for those SMEs only located at the same site, and a "4" to those SMEs located at different sites.

The strength of association between these two distance matrices and each of the two distance matrices and the consensus matrix was tested using Quadratic Assignment [29]. With Quadratic Assignment a correlation statistic γ is computed between the corresponding cells in two matrices of observed data. Then one of these matrices is repeatedly permuted randomly, each time computing a new γ. A p-value for this randomization test is determined by counting the proportion of times the value of γ computed for the data permutations equaled or exceeded the value calculated for the observed data. The results obtained from Quadratic Assignment testing with the Consensus matrix, and the Organizational and Locational distance matrices, after 1,000 permutations, are given in Table 4.

Table 4. Results from significance testing of relationships between organizational distance, inter-office distance and amount of consensus among SME responses. (Quadratic Assignment with 1,000 permutations used for tests.)

Association	γ (observed)	Proportion As Large
Organization/Location	0.586	0.000
Organization/Consensus	-0.388	0.810
Location/Consensus	-0.187	0.160

Several conclusions can be drawn from these tests. As one might expect, there does seem to be some association between a SME's organizational affiliation and the location of

his/her office. However, there exists little evidence to support the claim that either their organizational affiliation or the location of their office has much to do with what they know about "Online Privacy and Security," though location does seem to influence more what one knows than organizational affiliation, i.e., a possible "water cooler" effect. Another way of putting this is that cross-organizational forums and informal sharing of information among those who experience greater face-to-face contact may contribute more to learning and knowledge formation than hiring practices and interactions structured more strictly along organizational boundaries.

4. Conclusions

The experimental results obtained for the *Schemer* prototype are promising, especially considering that the "correct" answers obtained for the GVU sample were in many cases tentative due to a large, heterogeneous sample. Moreover, some of these answers were derived statistically, with no prior analysis to weed-out items with near equal frequencies of "true" and "false" responses. This finding implies that meaningful answers to difficult and "fuzzy" problems might be obtained more quickly, and with less effort and cost, from the information provided by a few competent SME's, rather than from a very much larger survey sample [13].

So, what does this experiment have to do with the Semantic Web? We believe that it demonstrates a potentially powerful use of consensus for deriving semantically-relevant ontologies from domain experts. While this experiment asked SMEs to evaluate items pertaining to Internet security and privacy, they might instead have been requested to rate terms in a list on the basis of their salience to a knowledge domain, or to rate the relative strength of semantic relationships between terms on this list. The protocol adopted for the present experiment could be applied to analyze SME responses to these items to determine (1) those terms that should be part of a controlled vocabulary, and (2) a standard set of semantic relationships between terms in this vocabulary. Based on these consensus views other items could be developed and evaluated by SMEs to derive defining attributes for terms in the ontology. At each step in this process, Consensus Analysis provides important "reality" checks. The metrics it yields, as computed in this study, more clearly indicate the saliency of the targeted domain to SMEs and provide an opportunity to assess how much domain knowledge is possessed by each SME in the sample. We conclude with the conjecture that, by interviewing even a small number of competent SMEs, ontologies for Web catalog and directory services can be similarly constructed in a manner that best represents the collective wisdom of semantically-specialized communities-of-interest.

5. Future Work

This study provides motivation for future research in four key areas: Information acquisition, knowledge derivation, knowledge representation and knowledge reuse.

5.1 Information Acquisition

The ubiquity of the Web is encouraging some in the Knowledge Management community to consider the automation of tried-and-tested information gathering techniques, e.g., repertory grids [30]. Many such techniques exist, and it isn't always clear when application of any particular one is appropriate, e.g., see [31, 32, 33]. Thus, there is a need to consider which of the many available information acquisition techniques are appropriate for gathering the information needed to derive different types of knowledge, e.g., controlled vocabularies, ontologies or production systems, and how best to deploy them electronically. In fact, a taxonomy of knowledge types, with a mapping of acquisition methods to each, is needed.

Fortunately, this study was able to reuse items developed for the *GVU Survey*. But this was only a proof-of-concept. Any meaningful implementation of the *Schemer* System (or another like it) will require support for item development, preferably by incrementally building pools from items submitted by SMEs. As in test development, items will have to be classified by their author's COI, then carefully pretested and analyzed for their discriminability before being added to an item pool. There is also an opportunity for evaluating alternative protocols for presenting items to subjects based on their background and the knowledge domain being tested, and more flexible highly-interactive formats for presenting items to subjects electronically.

5.2 Knowledge Derivation

Consensus Analysis provides a rigorous framework for deriving knowledge from information acquired from a group of SMEs. However, further refinements of the method are required to accommodate missing information and guessing, different difficulty levels of problems, and to enable appropriate classification of decision-makers into their respective communities-of-interest. These features are particularly important for supporting the idea of acquiring information from SME's incrementally and at their convenience. With regards to this last point, we envision the use of wireless communication devices, e.g., PDAs, as a useful means to acquire information from SMEs in less-intensive, asynchronous sessions.

5.3 Knowledge Representation

For derived knowledge to be useful, it needs to be represented in a way supported by other tools, but without sacrificing information about the details of its structure and semantics. Hence, the expressiveness and adequacy of existing knowledge representation standards, e.g., KIF [34], KRSL [35], RDF Core [2] and DAML+OIL [3] need to be reviewed and evaluated.

5.4 Knowledge Reuse

A minimal use of derived knowledge would be to publish it electronically. However, in the case of some knowledge, e.g., controlled vocabularies or ontologies, new services will be

required to support rapid integration of this knowledge with other technology. Thus, there is need to further explore new technologies for making knowledge more accessible to end-users and software applications.

6. Summary

This paper described the *Schemer* prototype, a Web-based infrastructure to acquire information from domain experts and process this information with a quantitative technique known as *Consensus Analysis.* This approach yields metrics that determine (1) whether a particular problem domain has salience for a group of subject matter experts, (2) the level of competency for each of the subject matter experts, (3) the consensus view of this group weighted by the competency of its members, and (4) a classification of subject matter experts by their appropriate community-of-interest.

There is an opportunity to harness the same social-cultural processes that fostered the creation, growth and success of the current Web to evolve rich ontologies. These will be the focal point of the "emergent" Semantic Web and will be constructed dynamically based on consensus processes. Distributed and semantically-rich information spaces, supported by the infrastructure needed to easily navigate them, promise to be the transforming technology of the 21st century. New knowledge derivation techniques, such as consensus analysis, embedded in tools that enable dynamic evolution of ontologies are a critical component of the semantic Web. We see the *Schemer* prototype as an important step in the long march towards realizing a semantic Web infrastructure.

Acknowledgements

We want to thank Mark Rosenstein, Jon Kettenring, Sid Dalal and anonymous reviewers for commenting on earlier drafts of this paper, and Kim Romney, Sue Weller and Steve Borgatti for many fruitful exchanges on Consensus Analysis and its formal foundations. We are also grateful to Tracy Mullen, Marek Fiuk and Chumki Basu for their assistance with implementing the *Schemer* prototype, and to other colleagues in Telcordia Technologies' *Information and Computer Sciences Research* and *Internet Architectures Research* labs for participating in the experiment described in this paper. Both Insightful *S-Plus*® version 5.0 and Analytic Technologies *AnthroPac*® version 4.1 were used to benchmark data analysis.

References

[1] Fensel, D., 2001. Understanding is based on Consensus. Panel on Semantics on the Web. *10th International WWW Conference, Hong Kong*, 2001.

[2] The RDF Core Working Group. http://www.w3.org/2001/sw/RDFCore/

[3] D. Broekstra et al., 2001. Enabling Knowledge Representation on the Web by extending the RDF Schema. *10th International WWW Conference, Hong Kong* 2001.

[4] T. H. Davenport and L. Prusak. 1998. Working Knowledge: How Organizations Manage What They Know. Boston: Harvard Business School Press.

[5] D. Gibson, J. Kleinberg, and P. Raghavan. 1998. Inferring web communities from link topology. *Proceedings of the 9th ACM Conference on Hypertext and Hypermedia.*

[6] W. Hill, L. Stead, M. Rosenstein, and G. Furnas. 1995. Recommending and evaluating choices in a virtual community of use. *Proceedings of the CHI-95 Conference, Denver, CO.*

[7] C. F. Cargill, 1989. Information Technology Standardization: Theory, Process, and Organizations. Bedford, MA: Digital Press.

[8] C. Cargill, 1994. Prologue and Introduction. *Standard View* **2(3)**: (1994) 129.

[9] A. Farquhar, R. Fikes, and J. Rice. 1996. The Ontolingua Server: A Tool for Collaborative Ontology Construction. Knowledge Systems Laboratory, KSL-96-26 (September).

[10] M. M. Turoff and S. R. Hiltz. 1996. Computer-based Delphi processes. In M. Adler and E. Ziglio (eds.), *Gazing into the Oracle: The Delphi Method and its Application to Social Policy and Public Health.* pp. 56-85. London: Jessica Kingsley Publishers.

[11] R. M. Cooke, 1991. *Experts in Uncertainty: Opinion and Subjective Probability in Science.* New York: Oxford University Press.

[12] H. A. Linstone and M. Turoff (eds.). 1975. *The Delphi Method: Technique and Applications.* Reading, MA: Addison-Wesley.

[13] A. K. Romney, S. C. Weller, and W. H. Batchelder. 1986. Culture as consensus: A theory of culture and informant accuracy. *American Anthropologist* **88(2):313-338.**

[14] R. M. Keesing, 1974. Theories of culture. *Annual Review of Anthropology* **3:73-97.**

[15] A. L. Kroeber, 1948. *Anthropology.* New York: Harcourt, Brace.

[16] D. W. Read and C. A. Behrens. 1989. Modeling folk knowledge as expert systems. *Anthropological Quarterly* **62(3):107-120.**

[17] D. W. Read and C. A. Behrens. 1991. Computer representation of cultural constructs: New research tools for the study of kinship systems. In M. S. Boone and J. J. Wood (eds.), *Computer Applications for Anthropologists.* Pp. 228-250. Belmont, CA: Wadsworth Publishing Co.

[18] FGDC. 1994. *Content Standards for Digital Geospatial Metadata (June 8).* Washington, D. C.: Federal Geographic Data Committee.

[19] B. J. Hillman, R. G. Swensson, S. J. Hessel, D. E. Gerson, and P. G. Herman. 1997. Improving diagnostic accuracy: A comparison of interactive and Delphi consultations. *Investigative Radiology* **12:112-115.**

[20] B. W. Boehm, P. Bose, E. Horowitz and M. J. Lee. 1994. Software requirements as negotiated win conditions. *Proceedings of Information CommunityRE (April)* Pp. 74-83.

[21] R. G. D'Andrade, 1981. The cultural part of cognition. *Cognitive Science* **5:179-195.**

[22] W. H. Batchelder and A. K. Romney. 1986. The statistical analysis of a general Condorcet model for dichotomous choice situations. In G. Grofman and G. Owen (eds.), *Information Pooling and Group Decision Making.* Pp. 103-112. Greenwich, CT: JAI Press.

[23] W. H. Batchelder and A. K. Romney. 1988. Test theory without an answer key. *Psychometrika* **53:71-92.**

[24] D. Caulkins and S. Hyatt. 1999. Using consensus analysis to measure cultural diversity in organizations and social movements. *Field Methods* **11(1): 5-26**.

[25] S. C. Weller and N. C. Mann. 1997. Assessing rater performance without a standard using consensus theory. *Medical Decision Making* **17:71-79**.

[26] A. L. Comrey, 1962. The minimum residual method of factor analysis. *Psychological Rep*orts **11:15-18**.

[27] S. S. Schiffman, M. L. Reynolds, and F. W. Young. 1981. *Introduction to Multidimensional Scaling: Theory, Methods and Applications.* New York: Academic Press.

[28] H. M. Blalock, Jr. 1972. *Social Statistics.* New York: McGraw-Hill.

[29] L. J. Hubert, 1987. *Assignment Methods in Combinatorial Data Analysis. Statistics, textbooks and monographs,* **(73)** New York: Marcel Dekker, Inc.

[30] J. H. Boose, 1989. A survey of knowledge acquisition techniques and tools. *Knowledge Acquisition* **1(1): 3-37**.

[31] H. R. Bernard, 1988. *Research Methods in Cultural Anthropology.* Newbury Park, CA: Sage.

[32] J. P. Spradley, 1979. *The Ethnographic Interview.* New York: Holt, Rinehart and Winston.

[33] O. Werner and G. M. Schoepfle. 1987. *Systematic Fieldwork* (2 vols). Newbury Park, CA. Sage.

[34] M. R. Genesereth and R. E. Fikes (eds.). 1992. *Knowledge Interchange Format, Version 3.0 Reference Manual.* Computer Science Department, Stanford University, Technical report Logic-92-1.

[35] J. Allen and N. Lehrer. 1992. *Knowledge Representation Specification Language (KRSL), Version 2.0.1 Reference Manual.* Draft of the DARPA/Rome Laboratory Planning and Scheduling Initiative. ISX Corporation.

Appendix

Items derived from Georgia Tech's Graphics, Visualization and Usability Center's *10th World Wide Web User Survey* on "Online Privacy and Security." Answers in parentheses based on simple "majority view" obtained from survey of 1,482 respondents. (See *http://www.gvu.gatech.edu/gvu/user_surveys/survey-1998-10/.*)

In general, how concerned are most WWW users about security on the Internet (e.g., others reading their email, finding out what websites they visit, etc.)? Keep in mind that in this context "security" can mean privacy, confidentiality, and/or proof of identity for a WWW user or for someone else.

1. Older (50+ years old) WWW users tend to be more concerned than younger users. (F)

2. Experienced (> 4 years experience) WWW users tend to be less concerned than inexperienced users. (T)

In general, how concerned are most WWW users about security in relation to making purchases or banking over the Internet? Keep in mind that "security" can mean privacy, confidentiality, and/or proof of identity for a WWW user or for someone else.

3. Older (50+ years old) WWW users tend to be more concerned than younger users. (F)

4. Experienced (> 4 years experience) WWW users tend to be less concerned than inexperienced users. (T)

One thing that makes it difficult to study Internet security is people's and business' reluctance to report security problems for fear of causing more problems for themselves. In addition, it is not always clear where they should be reported. One idea is to have a "clearinghouse" where security problems can be studied and tracked. Please provide you opinions about how such an idea might be received.

5. Most WWW users would be willing to report a security break-in of their personal machine or network to a clearinghouse that maintained their anonymity? (T)

6. Most WWW users would be willing to report a security break-in of their business machine or network to a clearinghouse that maintained their anonymity? (T)

7. Less than 10% of WWW users have ever had their credit card number stolen (either online or offline)? (F)

8. More than 50% of WWW users are willing to use their credit card on the web? (T)

9. Less than 20% of WWW users have an unlisted phone number? (F)

10. Most WWW users are unwilling to put their name and address in a directory for public access on the Web (e.g. the online equivalent of a phone company's "White Pages")? (F)

11. Most WWW users are willing to conduct banking on the Web without a statement from the bank of the security procedures used? (F)

12. WWW users within the United States are more concerned than those in Europe or elsewhere about conducting business online outside of their own country without a statement of the security procedures used? (T)

13. In general, PRIVACY is more important than CONVENIENCE to most WWW users? (T)

14. WWW users will more likely participate in an "online auction" for something they are interested in purchasing? (F)

15. Most WWW users think using the Internet for shopping and banking would make their life easier? (T)

16. For most WWW users security features are the deciding factor in choosing whether or not to do business with an Internet-based company? (F)

17. Most WWW users believe that metrics to measure "how secure" a specific site is rated would not be of any help or value to them? (F)

When one views a Web page, they issue a request to a machine that returns the page to them. Which of the following information do most WWW users believe is technically possible to record/log about their page request?

18. Their email address (T)
19. Time of the request (T)
20. Their machine address (T)
21. The requested page (T)
22. An identifier that persists across visits to that site (T)
23. The type of browser they are using (T)
24. Their machine's operating system (T)
25. Their geographical location (F)
26. Their screen size (F)

What information would most WWW users agree ought to be collected for each Web page they request?

27. Their email address (F)
28. Time of the request (T)
29. Their machine address (F)
30. The requested page (T)
31. An identifier that persists across visits to that site (F)
32. The type of browser they are using (F)
33. Their machine's operating system (F)
34. Their geographical location (F)
35. Their screen size (F)

Most WWW users would give demographic information to a Web site ...

36. if a statement was provided regarding what information was being collected (T)
37. if a statement was provided regarding how the information was going to be used (T)
38. in exchange for access to the pages on the Web site (F)
39. in exchange for a small discount at the Web site's store or on their products (F)
40. in exchange for some value-added service (e.g., notification of events, etc.) (F)
41. if the data would only be used in aggregate form (i.e., not on an individual basis) (T)

What conditions would cause most WWW users to refrain from filling out online registration forms at sites?

42. Takes too much time (T)
43. Required to give their name (F)
44. Required to give an email address (F)
45. Required to give their mailing address (F)
46. Information is not provided on how the data is going to be used (T)
47. Accessing the site is not worth revealing the requested information (T)
48. The entity collecting the data is not trusted (T)

Recent attention has been given to mass electronic mailings (a.k.a. spammings) which often contain advertisements, political statements, get-rich-quick schemes, etc. Among most WWW users, which of the following policies would most likely find support?

49. The Government ought to pass a law making it illegal. (F)
50. Mass mailing agencies ought to have to pay an 'impact' fee. (F)
51. A blacklist of spammers should be built to allow message filtering. (F)
52. A registry ought to be created which contains a list of those not wishing to receive mass mailings. (F)

53. Most of the time upon receiving a mass mailing, WWW users will read the message, then either send a message back asking not to be included in future mailings, retaliate in some manner (e.g., mailing bombings, denial of service, etc.), or perform some other action. (F)

Most WWW users would support which of the following?

54. New laws to protect privacy on the Internet. (T)
55. The establishment of key escrow encryption (where a trusted party keeps a key that can read encrypted messages). (T)
56. Web sites need information about their users to market their site to advertisers. (T)
57. Content providers have the right to resell information about its users to other companies. (F)
58. A user ought to have complete control over which sites get what demographic information. (T)
59. Magazines to which a WWW user subscribes have the right to sell their name and address to companies they feel will interest that user. (F)

60. WWW users like receiving mass postal mailings that were specifically targeted to their demographics. (F)
61. WWW users like receiving mass electronic mailings. (F)
62. WWW users ought to be able to take on different aliases/roles at different times on the Internet. (T)
63. WWW users value being able to visit sites on the Internet in an anonymous manner. (T)
64. WWW users ought to be able to communicate over the Internet without people being able to read the content. (T)
65. WWW users would prefer Internet payment systems that are anonymous to those that are user identified. (T)
66. Third party advertising agencies should be able to compile usage behavior across different web sites for direct marketing purposes. (F)
67. There ought to be stricter laws to protect children's privacy than adult's privacy on the Internet. (T)

Internet Applications and Standards

This page intentionally left blank

Real World Semantic Web Applications
V. Kashyap and L. Shklar (Eds.)
IOS Press, 2002

Change Detection of RDF Documents Using Signatures

Latifur KHAN, Lei WANG, and Qing CHEN
Department of Computer Science
University of Texas at Dallas
Richardson, TX 75083-0688
Email: [lkhan, leiwang, qingch]@utdallas.edu

Abstract. *Current web data is not machine-understandable. It is very hard to automate anything on the Web, and because of the volume of information the Web contains, it is not possible to manage it manually. The solution proposed here is to use metadata to describe the data contained on the Web. Resource Description Framework (RDF) is a foundation for processing metadata; the RDF model is simply a directed graph. The characteristics of RDF, and the property of being graph-structured (i.e. a collection of nodes), facilitate the detection of changes in a RDF document in minute detail and at a finer grain than that which is obtainable at the document level. There is at present no really effective method for detecting these changes. We wish to propose such a method here. Rather than inspecting all nodes between two versions of RDF documents, we propose an effective algorithm, called top-down, which will detect changes in RDF documents by exploring a subset of nodes in the tree built from an RDF graph. We would like to be certain that if a leaf node changes, the algorithm will detect these changes, not only by inspecting the node itself, but also its parent node, grand parent node, and so on. For this, a signature for each node will be constructed which is basically an abstraction of the information stored in the node. There are several ways we can construct such a signature. We will choose exclusive-or (XOR) to construct node signatures which will prevent a user from getting irrelevant information/change and to make certain that the user does not miss relevant information. Note that for the web, along with being able to access huge quantities of information, the relevancy of information/change is more important than the missing of relevant information/change. For this, in this paper, we propose an automatic change detection algorithm which will identify changes between two versions of an RDF document based on these signatures. Our proposed algorithm will traverse the least number of nodes necessary to detect these changes. We will demonstrate that our algorithm outperforms the traditional algorithm which exhaustively searches the entire space. We will also demonstrate analytically and empirically that the miss of relevant change is within tolerable range.*

1. Introduction

The availability of data on the web is constantly expanding. Most of this data is semi-structured, consisting of text (HTML) and multimedia (sound, video, image). Overall, this data constitutes the largest body of information ever accessible to any individual. The problem with this data is that it is not machine-understandable. It is very hard to automate anything on the Web, and because of the volume of information the Web contains it is not possible to manage it manually. The solution proposed here is to use metadata to describe the data contained on the Web [34].

Resource Description Framework (RDF) is a foundation for processing metadata which provides interoperability between applications that exchange machine-understandable information on the Web. RDF emphasizes facilities to enable automated processing of Web resources. RDF can be used in a variety of application areas; for example: in resource discovery to provide better search engine capabilities, and in cataloging for describing the content and content relationships available at a particular Web site, page, or digital library.

The RDF data model is a syntax-neutral way of representing RDF expressions. The basic data model consists of three object types, resources, properties, and statements. A RDF model can be represented by a directed graph. Therefore, any node in the RDF model can have multiple children and multiple parents.

Web users are not only interested in the current values of documents but may also be interested in their future modifications. Regarding these, we need to address the following two questions: First, we need to be able to detect the relevant changes in RDF documents. Second, it is necessary to notify the user effectively regarding the most recent changes. In this paper we address the first question. The second question will be addressed as a part of future work.

In approaching the first question, rather than inspecting all nodes between two versions of RDF documents, we propose an effective algorithm, called top-down, which will detect changes in RDF documents by exploring a subset of nodes in the graph. In other words, the top-down algorithm prunes the search space, starting by comparing values in the root nodes of the two versions. Next, immediate children nodes of the root nodes will be compared. In this procedure, we would like to make sure that if a leaf node changes, the algorithm can detect the change not only by inspecting the node itself but also its parent node, grand parent node, and so on. For this purpose, a signature for each node will be constructed. The signature is basically an abstraction of the information stored in each node. The signature of an interior node can be constructed using either OR or Exclusive-or (XOR) of all descendant's signatures. In IR for document filtering, the signature used is based on OR which may allow the retrieval of irrelevant documents, adversely affecting precision [28]. Furthermore, the number of nodes with 1 in the signature is assumed to be very small as compared to the length of the signature [30]. On the other hand, using XOR it is not possible to get any irrelevant information/change; however, some relevant information/change may be missed, adversely affecting recall. Note that in the web along with the availability of huge quantities of information, the relevancy of information/change (precision) is more important as compared to the loss of relevant information/change (recall). Therefore, we consider change detection of RDF documents using signatures based on XOR. Note that graph will be transformed into a tree in order to avoid further adverse effect on recall.

To reduce the number of nodes which need to be examined in comparing two versions, it will only be necessary to compare node signatures in the old version with corresponding node signatures in the new version. The top-down algorithm follows depth first search exploration.

The main contributions of this work are as follows: First, we propose an automatic change detection algorithm between two versions of RDF documents based on node signatures. Second, we use XOR to construct tree signatures. Due to the employment of XOR a user will not get any irrelevant change; however, the user may miss some change which is relevant. We demonstrate analytically and empirically, however, that this miss is very low. Finally, compared to other algorithms, our proposed algorithm traverses fewer nodes in the tree to detect changes between two versions. We also demonstrate that our algorithm outperforms the traditional algorithm, which must exhaustively search the entire space.

The remainder of this paper is organized as follows. Section 2 covers related work. Section 3 covers RDF basics. Section 4 presents our algorithm for detecting changes in RDF documents. Section 5 discusses merits and demerits of the usage of XOR for specifying node signatures and reports a simulation result regarding the use of XOR. Section 6 describes in detail the performance of our approach over an exhaustive search. Section 7 presents our conclusions, and a comment about future work.

2. Related Work

Recent work in change detection has focused on computing differences between flat and hierarchical files. We will first present related work related to flat files. Next, we will present related work related to hierarchical files.

The GNU diff utility and AT&Ts HtmlDiff [2, 9, 11], are examples of work related to flat files. GNU diff utility uses a LCS (Longest Common Subsequence) algorithm [19] to compare two plain text files. However, this approach cannot be applied to detect change in hierarchical files, like XML/RDF document.

AT&T [11] developed an Internet Difference Engine with HtmlDiff [2] to detect changes in two versions of an HTML page. In the HtmlDiff approach, they use tokens to represent sentence-breaking markup and sentence. Thus, two different versions of an HTML file can be viewed as two sequences of tokens. By comparing the weighted LCS, they can determine the best matching between two versions and identify the different parts.

To detect changes in hierarchical documents various methods are proposed based on tree-to-tree correction [3, 5, 8, 20, 26, 27]. A tree can be considered an ordered tree, which means the order of siblings is important, which implies that structural change detection has been addressed. Where an unordered tree is concerned, however, the change detection problem with an un-ordered tree is much more complicated. Zhang et al. point out that change detection in an unordered tree problem is in NP complete in the general case [28].

Chawathe et al. define the hierarchical change detection problem as the problem of finding a "minimum-cost edit script" that transforms one ordered tree to another, and they present an efficient algorithm for computing an edit script [5]. One major drawback of their method is that they assume that a leaf node in one version has in another version at most one good matching.

Cobena et al. treat an XML document as an ordered tree [8]. They devise an algorithm named Bottom-Up, Lazy-Down (BULD) propagation, and compute and propagate matching between common large sub-trees of the two documents.

Wang et al. present an algorithm for the unordered tree change detection problem [26]. They develop an algorithm in polynomial time complexity by exploring features of XML document which constitute a tree.

An RDF document is treated as an edge directed graph [32, 33, 34]. We transform the RDF graph into a tree structure by removing some edges without losing content information. Furthermore, each node can be uniquely identified between two versions. Therefore, structural change is trivial here (i.e., ordered/unordered tree). In this paper we address content-based change detection. Based on change, we propose an effective change detection algorithm by discarding irrelevant sub-trees based on the node signatures. Thus, we explore only a part of the tree, while the other parts are pruned.

Related work in information filtering uses signature-based technique which is mainly based on OR rather than XOR. Signature methods have been used extensively for text retrieval [28], image database [17], multimedia database [6, 21] and other conventional database systems [6].

A signature is an abstraction of the information stored in a record or file. By examining signatures, we can estimate whether the record contains the desired information.

3. RDF Primer

Resource Description Framework (RDF) is a foundation for processing metadata; it provides interoperability between applications that exchange machine-understandable information on the Web. RDF emphasizes facilities which enable the automated processing of Web resources [32]. Since RDF is designed to describe the resources and the relationship among them without assumption, the definition mechanism should be domain neutral, and can be applied to any domain. There are the following key terms in RDF [33]:

Resource: A Resource is anything that can have a URI (Unified Resource Identifier); this includes all the Web's pages, as well as individual elements of an RDF document. For instance, when we reach a webpage with hyperlink as hyperlink "http://weather.yahoo.com/regional/USTX.html"; that is a resource. Each resource is uniquely identified by a URI.

Property: A Property is a Resource that has a name and can be used as a property, for example, a company has many properties, such as its name and locations, etc.

Statement: A Statement consists of the combination of a Resource, a Property, and a value. It just like a simple sentence; Like, "The Company's main email address is :main@virtual.com." In this sentence, there are three parts:
The subject, company, which is a resource. The URI can be www.virtualsite.com
The predicate. The URI can be http://www.virturalsite.com/resource/email.
The object, which can be a resource or a literal value. Here the value:main@virturalsit.com.

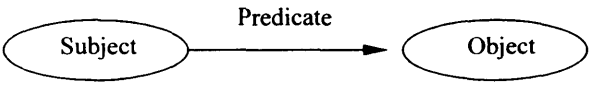


Figure 1. Basic model of RDF

We present this simple example to describe the basic model of RDF (refer to Figure 1). In Figure 2, we present a case which is a little bit more complicated. Say, there is a virtual company with the URI, http://www.virtualsite.com; The founder of the company has his own homepage, http://www.virtualsite.com/~qchen, and the full name of the founder is Qing, Chen; The main technical support email address is main@virtualsite.com; and the company introduction is composed by the founder. From these statements and relationships among them, we can draw the RDF model in Figure 3, and this RDF document is shown in Figure 2 (see [32] for more details).

From Figure 3, we see that a RDF model is an edge directed graph. A node in the graph represents a subject or an object, which can be URI, empty node or literal value; while the link from the subject to an object is the predicate, which can be a URI. Any node in the RDF model can have multiple children and multiple parents. For most cases, there are roots in the RDF

model, which do not have any parents. These points regarding the RDF model are very important for the design of our change detection algorithm.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:ex=" http://www.virtualsite.com/"
        xmlns:dc=" http://www.virtualsite.com/about/">
<rdf:Description rdf:about="http://www.virtualsite.com">
<ex:founder rdf:parseType="Resource">
   <ex:fullname>Qing Chen</ex:fullName>
   <ex:homePage rdf:resource=" http://www.virtualsite.com/~qchen" />
</ex:founder>
<ex:email>main@virtualsite.com<ex:email>
<ex:purpose rdf:resource=" http://www.virtualsite.com/about/index">
   <dc:creator rdf:resource=" http://www.virtualsite.com/~qchen"/>
</ex:purpose>
</rdf:Description>
</rdf:RDF>
```

Figure 2. An Example of RDF

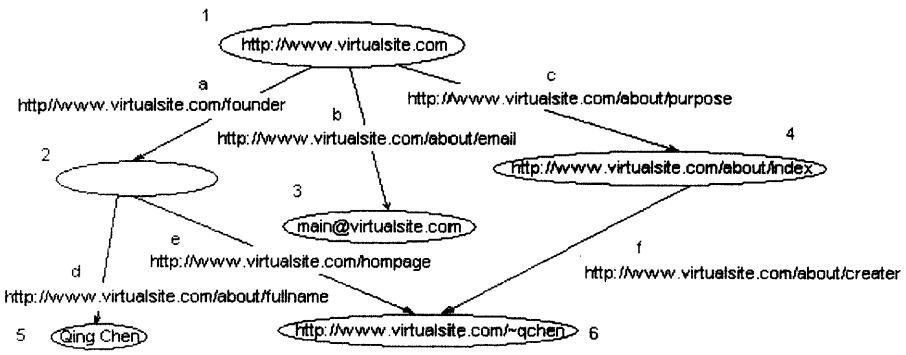


Figure 3. Model of RDF in Figure 1

4. Our Approach

As we know, the RDF graph (model) is a directed graph, and the model of the RDF consists of a group of statements. In the graph, a statement corresponds to nodes; i.e., one subject and one object, and the link which is directed from the subject node to the object node. For instance, in Figure 2, from node 1 there are three outgoing links, a, b, and c. If we go from node 1 through link a, we will be led to node 2. And this node1 (node2 relationship represents a statement with

subject URI "http://www.virtualsite.com" , while the object is an empty node with link a labeled "http://www.virtualsite.com/founder". So, if an RDF document changes, either some subjects or objects are changed, or some links' labels are changed. Our goal is to detect whether or not the two versions, before and after a change, are identical. If not we will match each node and label in the old version with its corresponding value in the new version in order to detect changes.

4.1 Naive Approach

In an RDF graph, the content of each internal node is either some URI or empty, and the content of each leaf node (the node with no child) is a literal value or a URI. Since URIs are globally unique, and for an empty node the system will generate a unique ID, each internal node can be assigned a unique key (either URI or a unique ID for an empty internal node) for matching. However, for link labels and leaf nodes the same content might appear many times in a RDF model. Fortunately, each statement contains a subject, an object and a predicate. Since the subject has a unique key, by combining the content of object and predicate we can easily generate a unique key for each statement. Thus, the signature of a statement is the function of the content of subject, predicate and object. If any content in a statement has been changed, the signature will be changed; and thus, we can detect the change.

One possible naive approach is to match each statement in the old version with its corresponding statement in the new version, based on signatures. Then the entire search space will be explored to detect changes, incurring high execution time. For example, if we apply the naive approach in Figure 2 to detect changes, we would have to traverse all the nodes in the graph. Consequently, we would like to devise an algorithm that can identify a sub-graph for further exploration while tossing out other sub-graphs if they do not show changes. Thus, the exploration of the search space can be reduced.

4.2 Top-Down Approach

In order to avoid checking all the statements in the RDF model, as in the naïve approach, we propose an effective method that only explores a subset of nodes in the RDF graph (see Figure 2) to detect change. The strategy incorporated in this approach is as follows: After getting a RDF graph based on the RDF model, we first calculate the content abstraction (Cabs) of each node based on the content of the node and its outgoing links and children. We then build a tree in the RDF graph (see Section 4.5). We next propose an effective algorithm for the detection of changes in the RDF by exploring only a subset of nodes in the constructed tree. This top-down algorithm starts by comparing values in the root nodes of the two versions. Next the immediate children nodes of the root nodes will be compared. In doing this, we would like to make sure that if a leaf node changes the algorithm can detect changes, not by inspecting the node itself, but also its parent node, grand parent node, and so on. For this, a signature for each node will be constructed. Therefore, we will not match every node in the first tree (old version) to every node in the second tree (new version). Note that each node in an RDF has a unique identifier. Exploring this identifier will help us to identify corresponding nodes between two RDF versions. In other words, in order to match each node in the older version with its corresponding node in the new version we need to use URI.

4.3 Graph Construction

In the RDF model, the RDF document is actually stored as a list of statements which cannot be traversed. In order to apply top-down detection it is necessary to build an RDF graph, based on the list of statements in the model, i.e., by replacing the subject and object relationships with the parent and child relationships among those nodes.

The algorithm for transferring an RDF model into a graph is not complicated. First, we maintain a node set which is initially empty. Next, we get the statements from the RDF model one by one. For each statement we will check and see if the subject and object exist in the node set with reference to the fact that each resource has a unique URI and each empty node has a system assigned id. If this is the case, we add the link as an outgoing link at the subject node. If it is not the case we will create a new node for the subject or object, or both, and add the link as an outgoing link at the subject node. The pseudo code for the building graph algorithm can be written as:

Definition 1: A node set N is a set of node and it is initially empty
Definition 2: A Link set, L for an internal node is all outgoing links from that node.

Graph Construction (model)

```
Node set N = null;
While (has next statement)
    If (subject ∈ N)
        Add link as subject's outgoing link in L;

        If (object ∉ N)
            Create a new node for object and add it into node set N;

    Else
        Create a new node for subject and add it into node set N;
        Create a new link set L for the subject and add link into L;

        If (object ∉ N)
            Create a new node for object and add it into node set N;

    Next statement;
```

Figure 4. Algorithm of Building Graph from a RDF Model

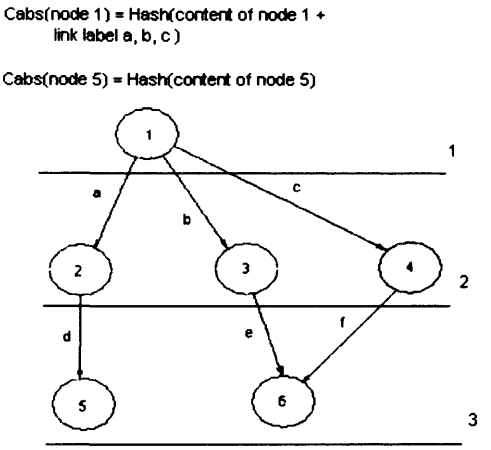


Figure 5. Calculation of Node Content Abstraction (Cabs)

4.4 Node Content Abstraction

We define node content abstraction (Cabs) in our top-down approach. We simplify Figure 4 into Figure 5 by using a shortened notation for clarity. By definition, Cabs (node Content abstraction), is the abstraction that stores all information concerning the contents of node itself and all the outgoing links. Thus, for an internal node, the content abstraction of this node will be the hash function of the node's content concatenating all its outgoing links' content. For a leaf node, the content abstraction will be the hash function of the content of leaf node itself.

For example, if we want to calculate the content abstraction of node 1 in Figure 3, Cabs (node1) will be concerned with the contents of node 1 and outgoing labels a, b, c. This implies that the node 1 Cabs is the hash function of the contents of node 1 and outgoing labels a, b, c. If any content changes, including a subject or object change, or link label change, the node 1 abstraction will be changed. The expression can be written as: Cabs (node 1) = Hash (content of node 1 + content of link label a, b, c). It is important to note that by calculating Cabs based on node content and outgoing link contents we can detect changes even at the link level.

If we want to calculate leaf node content abstraction, as in node 5, its abstraction is simply the function of the content of node 5 itself, which is Cabs (node 5) = Hash (content of node 5).

4.5 Tree Construction

We cannot apply Exclusive OR (XOR) to detect change in a graph directly, since there may be multiple ways to access a node in a graph if the node has multiple parents. For a node signature, we simply use XOR of all the Cabs of its descendant nodes along with its own Cabs. If a particular node, x, has more than one parent, the signature of x will participate in more than one of the signature constructions of an ancestor node (y) of its parent nodes. Note that the signature of a node is Ex-Or of its own Cabs and the signature of all its children nodes (see

Section 4.6). In this case, any change in x content will be unnoticed in the signature of y. If no descendant node of y has been changed except x, the signature of y will be the same. This is because Cabs(x) will participate twice in signature (y). Recall that Cabs(x) Ex-Or Cabs(x) = 0. For example, in Figure 5 node 6 has two parents, node 3 and node 4. Thus:

Sig (node 6) = Cabs (node 6)
Sig (node 3) = Cabs (node 3) XOR Sig (node 6) = Cabs (node 3) XOR Cabs (node 6)
Sig (node 4) = Cabs (node 4) XOR Sig (node 6) = Cabs (node 4) XOR Cabs (node 6)
Sig (node 1) = Cabs (node 1) XOR Sig(node 2) XOR Sig (node 3) XOR Sig (node4)
= ...Cabs (node 3) XOR Cabs (node 6) XOR Cabs (node 4) XOR Cabs (node 6)
= ... Cabs (node 3) XOR Cabs (node 4) XOR Cabs (node 6) XOR Cabs (node 6)

It is clear that Cabs (node 6) XOR Cabs (node 6) = 0. So, in this case, whether node 6 changes or not, there is no reflection at the root. To avoid this problem, we construct a tree structure, since each node in the tree has at most only one parent.

The basic idea of building a tree in the RDF graph is to remove some of the links in the graph so that each node in the RDF graph has only one parent at most. We first find the root of the RDF graph. By definition we establish that those nodes which have no parents can serve as roots, which means that in an RDF graph there might be more than one root. We then use a breadth first search (BFS) algorithm to build the tree.

Beginning with the root, we mark the root node as a node in level one, and all the children are marked as the nodes in level two and so on. To transform a graph into a tree, we follow a minimum spanning tree algorithm. For this, we need to assign a weight to each link and choose one as a minimum. For example, in Figure 3, node 1 is in level one, nodes 2, 3, 4 are in level two, and nodes 5, 6 are in level three. For each link label, we calculate a link weight based on the content of the link. From the level 1 node we explore their children. For each child there are two possibilities. The first possibility is the node that has been visited. The second possibility is the node that has not been visited.

With regard to the first possibility we compare the level number of the node. If the level number is equal to its parent's level number plus one, and the link weight is smaller than the previous minimum weight of the node, we remove the original minimum weight link, connect the node from this parent to the node, and update the minimum weight attribute of this node using this link's weight. Otherwise, we remove this link. For example, in Figure 3, considering node 4 in level 2, we note that it has a child node 6. However, when we go from node 4 to node 6, we find that node 6 has already been visited, and the link weight from node 4 to node 6 is greater than the previously assigned weight from node 3 to node 6. So, this link will be discarded. Thus, the graph in Figure 5 turns into a tree in Figure 6.

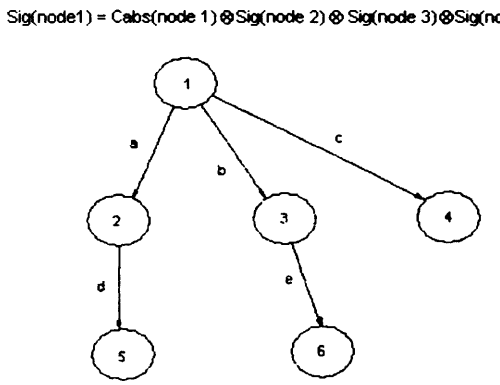


Figure 6. A Tree is Constructed by Tree Construction Algorithm

With regard to the second possibility we compare the child that has not been visited. For this, we simply connect the child, set the level number for the child node, calculate the link weight, and set this as the minimum weight for the child. This process is carried out recursively, until all the nodes in graph have been connected.

The pseudo code of the building tree algorithm can be written as:

Definition 3: Wij is the virtual link weight of link that connects node **i** to node **j**, i.e., Wij = F (content of the link going from node **i** to node **j**)

Definition 4: A root is an array list of nodes

Definition 5: A level is the level number of those nodes in the array list root

Tree Construction (root, level)

```
For each node Ni in the root
    If node Ni has children
        For each child node Nj of node Ni
            If ( isVisited (node Nj))
                If ((level(nodej)=level-1)
                and (minimum_weight(nj) > Wij))
                    Remove the link which has the minimum weight
                    minimum_weight (Nj)=Wij;
                Else
                    Remove this link from Ni to Nj
        Else
            Link node i to j;
            Calculate Wij;
            Minimum_weight (Nj)=Wij;
            Nj.level=level;
            Add node i to temporary array list temp;

root=temp;
```

```
level++;

Tree Construction (root, level);
```

Figure 7. Algorithm of Constructing a Tree for a RDF Graph

4.6 Node Signature

To reduce the number of node comparisons between two versions, we will compare the node signature in the old version with its corresponding node signature in the new version. The node signature will reflect all the sub-tree information, including nodes and links, and will store this sub-tree abstraction at the root of the sub-tree. Furthermore, the signature of interior node is simply Exclusive Or (XOR) of the Cabs (Content Abstraction) of the node itself and all its children nodes' signatures. Rather than using a concatenation of signatures we use XOR to reduce the size of the signature (see Section 5 for details). Formally, the definition of sub-tree signature is as follows:

Suppose x is a node in a tree T, Signature (node x) =Cabs (node x) XOR Signature (node x1) XOR Signature (node x2) ... XOR Signature (node xn), where x1, x2, x3, , and xn are the descendents of x. If x is a leaf node, Signature (node x) = Cabs (node x) = Hash (content of node x). For example, signature of the tree in Figure 3 is Signature (node 1) = Cabs (node 1) XOR Signature (node 2) XOR Signature (node 3) XOR Signature (node 4), and this signature will be stored at the root of the tree, node 1.

4.7 Change Detection

For change detection, we will first identify whether there is any change between two versions. If the answer is yes, we will determine where this change occurs. To make this determination, the signatures of two root nodes of two versions are compared. If these signatures (root signature) are the same there is no change between documents. Recall that the signature of the root includes all the information in the tree. If there is a change in any descendent node, the signature of its intermediate parent node will be changed, and thus that of the root node. If root signatures are different we need to address the problem of comparing nodes. For this, the signature of the root's children nodes in the old version will first be compared with its corresponding node signature in the new version. If these are the same, sub-trees rooted by these nodes from the two versions will be discarded. In other words, no further comparison will take place for these two nodes in the two versions and their descendents. When the node signatures are different, but all the children node signatures are matched and identical in the two versions, we will have arrived at the nodes where changes appear. If we catch the changed node, i.e. meaning that the node's content abstraction is changed, we can declare that at least a statement starting with this node has changed (referring to the node signature definition). Furthermore, we can then determine what kind of change has taken place, a subject or an object change, or a predicate change.

For example, let us consider the tree we have built in Figure 6. We assume that the link from node 3 to node 6 has been changed. Since this link is the outgoing link of node 3, the Cabs of node 3 will be different in two versions. Based on Figure 6 we calculate the signature of root and its children nodes. As node 3 has been changed, its signature, and ancestor nodes'

signature will also be changed. Thus, we compare root signature between two versions. Then we compare all its children nodes' signatures due to root signature difference. Thus, we determine that the change is in the statement starting from node 3, and hence, we identify the link change. Since no other node has been changed, the signatures of node 2, and 4 will be the same. Thus, no other sub-tree will be traversed for further change detection.

Definition 6: A node set N1 is a set containing the node for the old RDF model
Definition 7: A node set N2 is a set containing the node for the new RDF model
The pseudo code of TD change detection algorithm can be expressed as:

```
Change Detection (root, N1, N2)

While (root!=null)
    Boolean catch = true;
    For each node ni, in the root, N1
        Find corresponding node nj in N2;
        If (signature(ni)=signature(nj))
            This subtree has not been changed
        Else
            For each ni's child nk
                Find corresponding node nl in N2
                If (signature(nk)=signature(nl))
                Else
                    Add ni's child into root;
                    Catch=false;
            If (catch)
                Catch one changed node
```

Figure 8. Algorithm of Top-down Change Detection

5. Merits and Demerits of the Usage of XOR

We can get the signature of a parent node by superimposing signatures of all its children using XOR. If some bits change in a parent's signature, we can say that there must be some changes in the children's signatures. But conversely, if some children's signatures change, the parent's signature may remain the same. In this case, we will not be able to detect changes. We call this miss drop. This may happen because of the employment of XOR. For example, if a node has two children with signatures 1101, and both of these signatures are changed to 0100 at the same time (last bit changes in both cases), we will not be able to detect the underlying change. This is because 1101 XOR 1101 = 0100 XOR 0100.

In IR for document filtering, the node signature is used based on OR [24]. OR may allow to retrieve irrelevant documents to query (known as false drop) that affects precision. Furthermore, the following assumption is made: the number of 1 in the signature can be assumed to be very small as compared to length of the signature [16]. On the other hand, using XOR we will not get any irrelevant change (i.e., false drop probability = 0). While we may miss some change that affects recall; precision will not be affected.

During search or change detection, we want a user to get little irrelevant information/change (precision) while at the same time relevant information will not be overlooked (recall). Note that in the web environment precision is more important than recall. This is because the user wants to get only relevant information/change from the vast quantities of information that are being disseminated. For this, in order to discard irrelevant information, the user may sacrifice some relevant information/change. Furthermore, for web recall cannot be calculated directly [1]. Therefore, we would like to consider change detection of RDF documents in a web environment using XOR. XOR guarantees that precision will not be degraded, something which may happen using OR. Furthermore, it does not make any assumption about the least number of 1 in the signature.

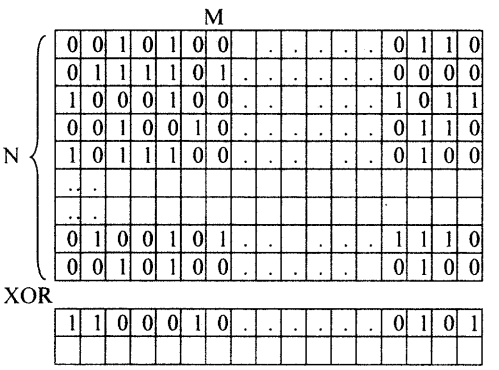


Figure 9. N Signatures with Length M Merge

Now, we will discuss when our approach fails to detect changes. Let us define that the length of signature be M and we have N signatures to superimpose together. In other words, the parent node has N children nodes. Then we have a bit pattern with N rows and M columns as Figure 9 and the value of each bit is either one or zero. It is obvious the parent's signature has nothing to do with the order of XOR operations. The only factor that can affect the final result is the number of 1 bit in each column. If this number in one column is even, we have 0 at the corresponding bit in the parent's signature. Otherwise, if this number is odd, we have 1. So, if we have one bit changed in a column, no matter what content changes (i.e., from 1 to 0 or 0 to 1), the parents signature in this column will be changed. If we have two bits changed in a column, the total number of 1 in the column will be either even or odd as before, the parent's signature in the respective bit will not be changed. Therefore, if the number of changed bits in every column is even (i.e., all columns), the value of each bit in parent's signature will not be changed. If this happens, we cannot detect the changes in children's signature from their parent's signature, which contributes miss drop. Otherwise, if the number of changed bits in any column is odd, at least this column in parent signature will be changed. So, we can detect the change.

5.1 Calculation of Miss Drop Probability

We assume a particular node with N descendant nodes. In addition, the signature length of a node is M. Therefore, a particular node signature will be simply XOR of all descendant node signatures. In Figure 9 we show a two-dimensional table where one dimension represents M and the other dimension represents N. Therefore, we have total MN bits, any bit can be changed from MN. (i.e., uniformly distributed)

Table 1. Symbol Definitions

Symbol	Definition
P	Miss drop probability (Similar to $P_i(M)$, but takes N into account.)
M	Signature length
N	Total number of signatures
p	Percentage of changed bits
i	Total number of changed bits
B_j	Total number of changed bits in the j^{th} column
$P_i(M)$	Probability that B_j is even (include zero) for column j when the total number of changed bit is i with M columns (ignoring N); all column changed bits are even; where j varies from 1 to M.
Q_i	Probability of having total number of changed bit is I
S_i	Probability that all bits changed in at least one column when the total number of changed bits is i
$S_L(k)$	Probability that all bits changed in exactly k columns when $LN \le i < (L+1)N$
L	At most all bits changed in L columns; L>=k
Q	At least all bits changed in q columns

M, N, and p can affect the miss drop probability, P. However, in some case N will not affect P. When i < N, then $p = \frac{i}{MN} \Rightarrow p < \frac{1}{M}$. Intuitively, when the total number of changed bits is less than N, it is impossible to change all bits in any column. In this case, N does not have any impact on P; only M and p can affect P. P will be calculated using the following way:

First, let's define $P_i(M)$ is the miss drop probability when total changed bit number is i with M columns. Obviously, we can have expressions as below.

$$P_0(M) = 1 \quad (1)$$

$$P_2(M) = \frac{1}{M} \quad (2)$$

$$P_i(M) = 0 \text{ when i is odd} \quad (3)$$

When i is even and i> 2, $P_i(M)$ =

$$\frac{1}{M} \times P_{i-2}(M) + [\frac{M-1}{M} \times \frac{2}{M} \times \frac{1}{M}] \times P_{i-4}(M)$$

$$= \frac{1}{M} \times P_{i-2}(M) + \frac{2 \times (M-1)}{M^3} \times P_{i-4}(M) \quad (4)$$

Since all changed bits are uniformly distributed across the columns, i changed bits will be uniformly distributed in M columns. For Equation 1 when i=0, B_j=0 for all j, $P_0(M)$ = 1 that means probability of 0 bit changed (i.e., even) in any column is 1. For Equation 2, when only two bits are changed, the column of the first bit does not matter; however the column of the second bit is important. The second bit may appear in the same column as the first bit or not.

Since changed bits are distributed uniformly, the probability that 2 bits are in the same column (i.e., second bit will appear in the same column as the first bit), $P_2(M) = \frac{1}{M}$; the probability that both are not in the same column is $= 1 - P_2(M) = \frac{M-1}{M}$.

For Equation 3, if i is odd, no matter how these i bits are distributed, at least for one j, B_j is odd. Hence, number of changed bits in at least one column is odd that allows us to detect changes Recall that node changes will be unnoticed if all column changes are even including zero. Therefore, for all odd i, $P_i(M) = 0$.

For Equation 4, when i>2 we have two cases. Let us assume changed bits are picked one by one. In case 1 the first two bits are in the same column with probability $\frac{1}{M}$; and in case 2 the first two bits are in different columns with probability $\frac{M-1}{M}$. In case 1, we would like to make B_j even for all columns; the first two bits will appear into the same column; rest of i-2 bits will satisfy the same requirement with probability $P_{i-2}(M)$. In case 2, at least two more bits from the rest of i-2 bits will be distributed in these two columns to make B_j even for all columns. Besides these 4 bits, rest i-4 bits will make B_j even for all j with probability $P_{i-4}(M)$.

Now, we can generalize the following way:

$$P_0(M) = 1$$

$$P_2(M) = \frac{1}{M}$$

$$P_4(M) = \frac{1}{M} \times P_2(M) + \frac{2 \times (M-1)}{M^3} \times P_0(M)$$

$$= \frac{3M-2}{M^3} = c_{41} M^{-2} + c_{42} M^{-3}$$

$$P_6(M) = \frac{1}{M} \times P_4(M) + \frac{2 \times (M-1)}{M^3} \times P_2(M)$$

$$= \frac{5M-4}{M^4} = c_{61} M^{-3} + c_{62} M^{-4}$$

$$P_8(M) = \frac{1}{M} \times P_6(M) + \frac{2 \times (M-1)}{M^3} \times P_4(M)$$

$$= \frac{11M^2 - 14M + 4}{M^6}$$

$$= c_{81} M^{-4} + c_{82} M^{-5} + c_{83} M^{-6}$$

.....

$$P_i(M) = \frac{1}{M} \times P_{i-2}(M) + \frac{2 \times (M-1)}{M^3} \times P_{i-4}(M)$$

$$= c_{i1} M^{-\frac{i}{2}} + c_{i2} M^{-\frac{i}{2}-1} + c_{i3} M^{-\frac{i}{2}-2} \quad (5)$$

Finally, miss drop probability will be sum of all these $P_i(M)$. Recall that miss drop probability arises when number of changed bits in each column is even. Thus, we miss drop probability for any M and N:

$$P = \sum_{i=1}^{MN} P_i(M) \times Q_i \quad (6)$$

If i > N, it would be possible that every bit in one column will be changed. Thus, N will affect on P. However, when p is low, even if i > N, we can still use Equation 5. This is because changed bits are uniformly distributed across columns with low p, S_i (i.e., the probability of having at least one column in which every bit will be changed) may be very low. In other words, N can affect P only when every bit in at least one column changed. Therefore, N has very little effect to P when p is small. When M and N are fixed along with large p, then i and S_i will be increased. Furthermore, N will play more impact on P.

Now, we would like to define $S_L(k)$ probability of every bit in exact k columns will be changed when $LN \leq i < (L + 1)N$. We have two boundary values here.

- When $i < N$, $L = 0$, then $S_i = 0$, $S_L(0) = 1$, $S_L(k) = 0$ $(0 < k \leq M)$;
- If $(N-1)*M < i \leq MN$, $S_i = \sum_{k=q}^{L} S_L(k) = 1$, $S_L(k) = 0$ $(0 \leq k < q)$. This is because i > (N - 1) * M means that at least in one column every bit will be changed. Hence, q is equal to i – (N - 1) * M in this case. Otherwise q is equal to zero.

We have $\binom{MN}{i}$ total different bit patterns, and we assume probability for each pattern is the same due to uniform distribution. Based on probability theory, for all other i between N and (N-1)*M, we have

$$S_L(L) = \frac{\binom{M}{L} \times \binom{MN - LN}{i - LN}}{\binom{MN}{i}} \tag{7}$$

$$S_L(L\text{-}1) = \frac{\binom{M}{L-1} \times \binom{MN - (L-1)N}{i - (L-1)N}}{\binom{MN}{i}} - S_L(L) \times \binom{L}{L-1}$$

$$S_L(L\text{-}2) = \frac{\binom{M}{L-2} \times \binom{MN - (L-2)N}{i - (L-2)N}}{\binom{MN}{i}} - S_L(L-1) \times \binom{L-1}{L-2} - S_L(L) \times \binom{L}{L-2}$$

$$\cdots$$

$$S_L(k) = \frac{\binom{M}{k} \times \binom{MN - kN}{i - kN}}{\binom{MN}{i}} - \sum_{l=k+1}^{L} S_L(l) \times \binom{l}{k} \tag{8}$$

Using Equation 7 and 8, we get all S_L recursively. For example, if L = 1, we have $S_L(1)$ using Equation 7:

$$S_L(1) = \frac{\binom{M}{1} \times \binom{(M-1) \times N}{i - N}}{\binom{MN}{i}} = \frac{M \times \prod_{l=0}^{N-1} (i - l)}{\prod_{l=0}^{N-1} (MN - l)}$$

And then, we get $S_L(0)$ using Equation 8.

$$S_L(0) = \frac{\binom{M}{0}\times\binom{MN}{i} - \binom{M}{1}\times\binom{MN-N}{i-N}}{\binom{MN}{i}}$$

$$= 1 - \frac{\binom{M}{1}\times\binom{(M-1)\times N}{i-N}}{\binom{MN}{i}} \quad (9)$$

$$S_i = \sum_{k=1}^{L} S_L(k) = 1 - S_L(0)$$

$$= \sum_{k=1}^{L}\left[\frac{\binom{M}{k}\times\binom{MN-kN}{i-kN}}{\binom{MN}{i}} - \sum_{l=k+1}^{L} S_L(l)\times\binom{l}{k}\right] \quad (10)$$

When N is odd, miss drop probability P is as below:

$$P = S_L(0)\times P_i(M) = \left\{1 - \sum_{k=1}^{L}\left[\frac{\binom{M}{k}\times\binom{MN-kN}{i-kN}}{\binom{MN}{i}} - \sum_{l=k+1}^{L} S_L(l)\times\binom{l}{k}\right]\right\}\times P_i(M) \quad (11)$$

When N is even:

$$P = S_L(q)\times P_{i-qN}(M-q) + S_L(q+1)\times P_{i-(q+1)N}(M-q-1) + \ldots + S_L(L)\times P_{i-LN}(M-L)$$

$$= \sum_{l=q}^{L} S_L(l)\times P_{i-lN}(M-l) \quad (12)$$

5.2 Simulation Results of Miss Drop Probability

We calculate miss drop probabilities based on M, N and p. As we are not sure that how many signatures will be changed and how many bits in a signature will be changed, we would like to do a simulation study. Thus, we can only vary p, M and N. M varies from 4 to 16 increased by 4. N varies from 5 to 65 increased by 5 and p varies from 10% to 100% increased by 10%. In simulation miss drop probability has been calculated for every fixed M, N and p.

For simulation, first we have determined the number of changed bits i (assume even). Then we uniformly have distributed i changed bits across M columns and recorded how many changes happened in each column. If a certain column has already had N changes, then distribute it in some other column randomly that has less than N number of changes. Recall that a column has only N number of bits, and maximum possible bits changed in a column is N. After distribution, we have counted total number of changed bits in every column. If this number is even for every column, then miss drop has occurred. We have repeated the above steps ten million times and observed how many miss drops we have.

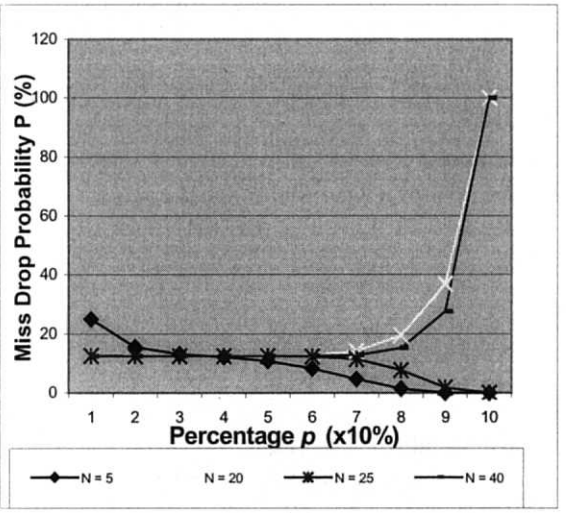


Figure 10. Miss Drop Probability, P for Different p (Fixed M)

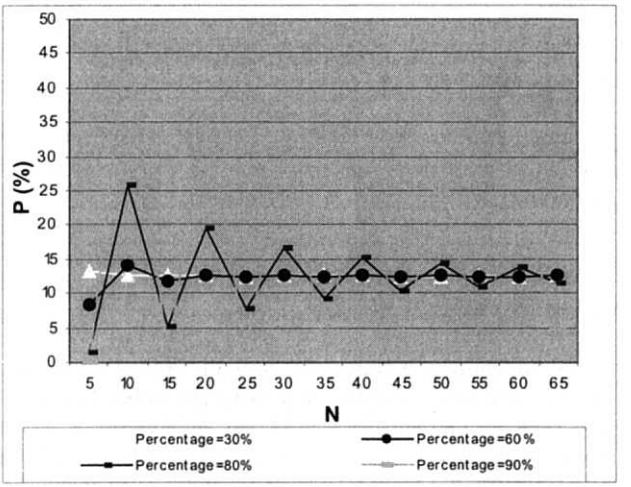


Figure 11. Miss Drop Probability, P for Different N (Fixed M)

In Figure 10, 11, 12 and 13, the X-axis represents p, N, M and M respectively. The Y-axis represents miss drop probability, P. In Figure 10 we demonstrate for different N how miss drop probability varies along with p when M is fixed (=4). We have observed P increases with the increased value of p when N is even (e.g., N=10, 20...). However, P decreases with the increased value of p when N is odd (e.g., N=5, 15). This is because N can affect P only when S_i is not equal to zero. If S_i is large, N will take more effect on P. From Equation 11 and 12, we know that the increase of S_i will also increase P when N is even; if N is odd, the increase of S_i will cause a decrement of P. From Equation 9 and 10, we know that S_i will increase with the

increased value of p. So, P will increase with the increased value of p when N is even and decrease when N is odd. In Figure 11, we have observed that P decreases with the increased value of N when N is even. On the other hand, when N is odd P increases with the increased value of N. This is because that S_i decreases along with the increased value of N when M and p are fixed.

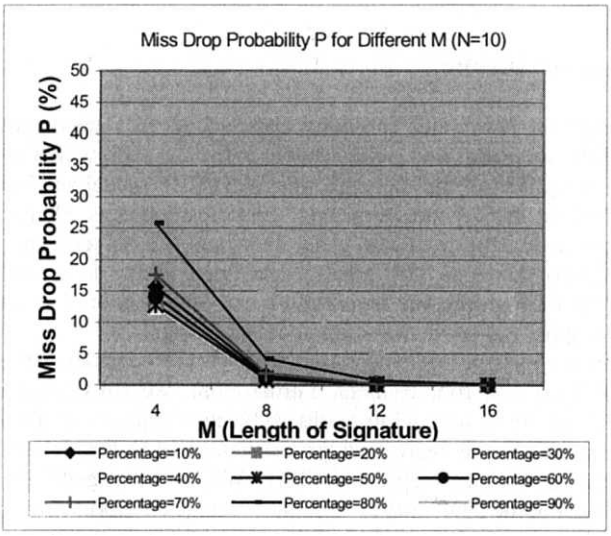


Figure 12. Miss Drop Probability P for Different M (Fixed N)

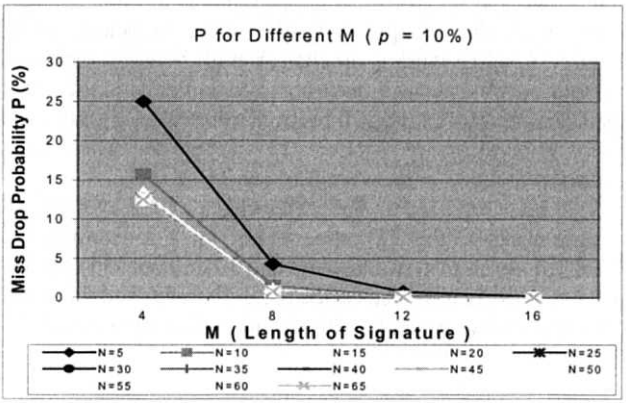


Figure 13. Miss Drop Probability P for Different M (Fixed p)

In Figure 12 and 13 we show how miss drop probability varies with M for fixed N and different p, and for fixed p and different N respectively. The data demonstrates that P decreases with increased value of M for any N and p. From Equation 9 and 10, S_i will decrease with the increased value of M. That implies that P should increase with the increased value of M when

N is odd. But in fact, according to Equation 5, 11 and 12, the other item will decrease sharply with the increased value of M. This will overcome the N's effect. So, no matter N is even or odd, P will decrease sharply with the increased value of M. In our simulation, when M = 16, P will be less than 0.01% for most N and p. In our implementation, Java provides 32 bit signatures which suggest that P will be insignificant. In other words, in web setting a tolerably small number of changes may be unnoticed.

6. Implementation and Results

In order to implement the Naive and Top-down approach of RDF document change detection, we first load the RDF document into a RDF model with Jena API, which is defined by the HP group [31]. Next, we gather all the statements from the RDF model and build an RDF graph , as in Figure 3, based on subject, predicate and object relationships. Hence, the subject is the source node (parent node), the predicate is the link, and the object is the destination (child node). When we have obtained an RDF graph we calculate all the node Cabs (see Section 4.3), locate the root of the RDF graph, and then build a tree (see Section 4.4). After calculating all node signatures, we apply top-down detection to detect changes.

Note that in our current implementation without loss of generality we assume that the RDF graph has at least one root element. Furthermore, we considered a very large RDF document, 336k. There are in total 4346 nodes (subjects + objects) in the RDF graph. Figure 14, and 15 show that the X-axis represents the percentage of leaf node changed for the Naive Approach (NA) and the Top-down approach (TD). We have changed 25%, 50%, and 75% of the leaf nodes changed in the RDF graph to simulate change environment.

In Figure 14 we have reported the performance of TD over NV in terms of number of nodes traversed (Y-axis). In TD we only need to explore a small number of nodes in the RDF graph to catch all the changes, while in the Naive approach it is necessary to check all the nodes to achieve the same goal regardless of the percentage of node changes. For this, the NV curve does not vary with the changes, i.e., the curve is nearly straight in Figure 14. Furthermore, in Figure 14, when there is no change at all between two versions there are still more than two hundred nodes that have been checked in the TD case. That is because in the original RDF graph there are more than 200 roots. Furthermore, if only one leaf node changes between two versions it is necessary to check 221 nodes, and when there are 16 leaf node changes (<25% leaf node change), we need to traverse 251 nodes.

In Figure 15 we have reported on the performance of TD over NV in terms of response time (Y-axis). We have observed that TD outperforms NV. For example, if there is a 50% leaf node change, NV and TD observe 50 millisecond and 30 millisecond respectively.

In Jena API, a method is provided called the difference model (model m) to detect RDF changes. Their approach mechanism is the same as our Naive approach [31].

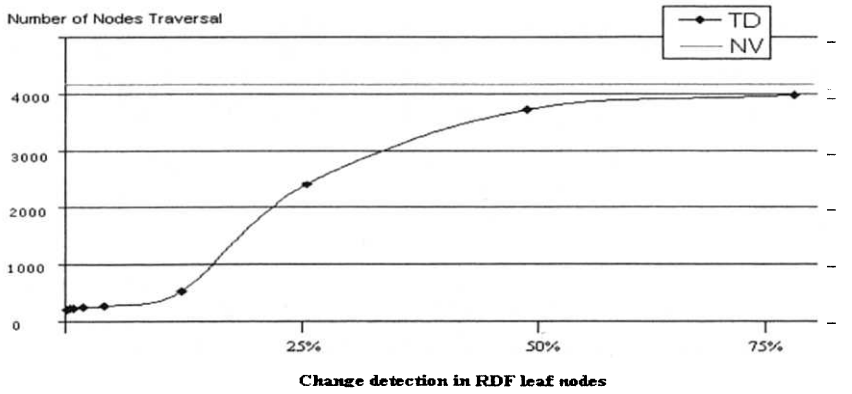


Figure 14. Comparison of NV and TD Approach for Number of Node Traversal

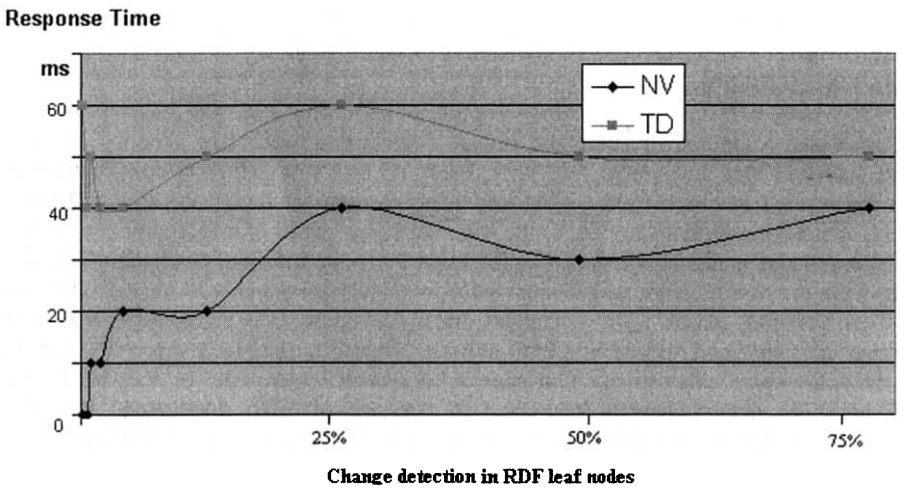


Figure 15. Comparison of NV and TD Approach for Response Time

In Figure 16 we have reported the performance of TD in terms of miss drop for leaf nodes (Y axis). Here, miss drop is low which is less than 1%, only about 0.6%. For example, for 40% leaf node change in RDF (337 nodes), only 0.6% of this change will be unnoticed (0.006*337 (2). Hence, miss drop is very low regardless of the percentage of node changes. Therefore, recall will not be hurt substantially. In addition, no node has been declared falsely as a content changed node. Therefore, precision will not be hurt at all.

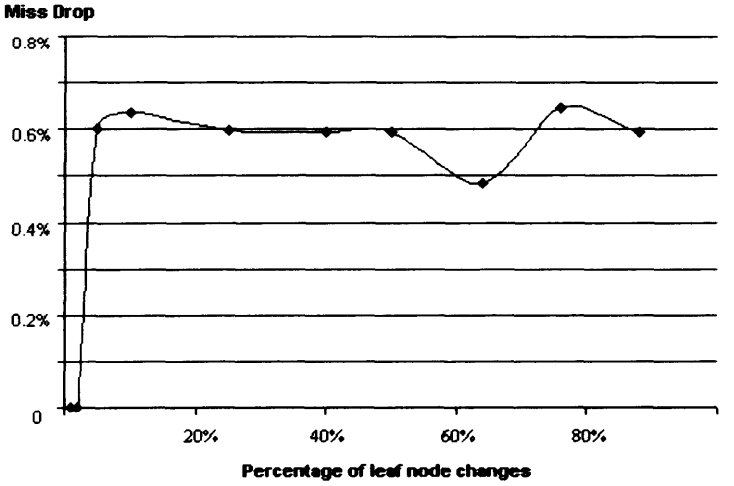


Figure 16. Miss Drop Probability for Leaf Nodes in TD Method

7. Conclusions and Future Work

We have proposed a method of automatic change detection between two versions of RDF documents based on Tree signatures. We have chosen exclusive-or (XOR) to construct a tree signature which prevents a user from getting irrelevant information/change and allows the loss of some relevant information. Note that for the web, with huge banks of information, relevancy of information/change is more important than the loss of relevant information/change. For this, in this paper we propose an automatic change detection algorithm which will identify changes between two versions of an RDF document based on these tree signatures. We have demonstrated that our novel approach is fast, and can outperform the Naive Approach, which searches the entire space exhaustively. Our approach traverses fewer nodes in trees built from the RDF graph to detect changes between two versions of RDF documents. We also demonstrate that the loss of relevant change in the web environment is within tolerable range.

We would like to extend this work in the following directions. First, we would like to store an old version of an RDF document into a database and then evaluate the impact of the database on change detection. Next, we would like to modify the algorithm so that it can be applied under all conditions, including cases in which there is no root in the RDF graph. Finally, we would like to address the question of how we can continually notify users of ongoing changes.

References

[1] R. Baeza-Yates, B. Ribeiro-Neto, Modern Information Retrieval, ISBN 0-201-39829.
[2] E. Berk, "HtmlDiff: A Differencing Tool for HTML Documents," Student Project, Princeton Univserity, http://www.htmldiff.com
[3] S. Chawathe, "Comparing Hierachical Data in External Memory", in Proc. of the twenty second International Conference on Very Large Data Base, VLDB, Bombay, India, 1996.
[4] S. Chawathe and H. Garcia-Molina, "Meaningful Change Detection in Structured Data," in Proc. of ACM SIGMOD International Conference on Management of Data, SIGMOD, pages 26–37, Tuscon, Arizona, May 1997.
[5] S. Chawathe, A. Rajaraman, H. Garcia-Molina and J. Widom, "Change Detection in Hierarchically Structured Information," in Proc. of the ACM SIGMOD International Conference on Management of Data, Montreal, June 1996.
[6] W.W. Chang and H.J. Schek, "A Signature Access Methods for the Starburst Database System," in Proc. of the Fifteenth International Conference on Very Large Data Base, Amsterdam, Netherlands, August 1989, pp. 145-153.
[7] Q. Chen, L. Khan, and L. Wang, "Content-based Change Detection of RDF Documents Using Signatures," submitted to International Workshop on Multimedia Information Systems, Tempe, Arizona, 2002.
[8] G. Cobena, S. Abiteboul, and A. Marian, "Detecting Changes in XML Documents", Proc. of the 18th International Conference on Data Engineering, ICDE, San Jose, California, March, 2002.
[9] F. P. Curbera, D. A. Epstein, "Fast Difference and Update of XML Documents" XTech'99, San Jose, March 1999
[10] F. Douglis, and T. Ball "Tracking and Viewing Changes on the Web", 1996 USENIX Annual Technical Conference, 1996.
[11] F. Douglis, T. Ball, Y. F. Chen, and E.Lpitspfois, "The AT&T Internet Difference Engine: Tracking and Viewing Changes on the Web", World Wide Web, 1(1): 27-44, Jan. 1998.
[12] C. Faloutsos and S. Christodoulakis, "Signature Files: An Access Method for Documents and its Analytical Performance Evaluation," ACM Trans. on Office Information Systems (TOOIS), 2, 4, pp. 267-288, Oct. 1984.
[13] C. M. Hoffmann, and M. J. O' Donnell, "Patten Matching in Trees", Journal of the ACM, 29: 68-95, 1982
[14] L. Khan, Q. Chen and Y. Rao, "A Comparative Study of Storing XML Data in Relational Database Management Systems," International Conference on Internet Computing (ICC), Las Vegas, June 2002.
[15] L. Khan, L. Wang, and Y. Rao, "Change Detection in XML Documents Using Signature-based Technique," International Workshop Real World RDF And Semantic Web Applications, in conjunction with WWW Conference, Hawaii, May 2002.
[16] H. Kitagawa, Y. Fukushima, Y. Ishikawa and N. Ohbo, "Estimation of False Drops in Set-valued Object Retrieval with Signature Files," in Proc. of 4th Intl. Conf. on Foundations of Data Organization and Algorithms, Chicago, Illinois, October 1993, Splinger-Verlag, pp. 146-163.
[17] S.-Y. Lee, M.-C. Yang, and J.-W. Chen, "Signature File as a Spatial Filter for Iconic Image Database," Journal of Visual Languages and Computing, vol. 3, no. 4, pp. 373-397, 1992.
[18] H. Maruyama, K. Tamura and R. Uramoto, "Digest values for DOM (DOMHash) Proposal", IBM Tokyo Research Lab, http://www.trl.ibm.co.jp/projects/xml/domhash.htm, 1998.
[19] E. W. Myers, "An O(ND) Difference Algorithm and Its Variations", Algorithmica, 1(2): 251-266, 1986.
[20] B. Nguyen, S. Abiteboul, G. Cobena, and M. Preda, "Monitoring XML Data on the Web," in Proc. of the ACM SIGMOD, Santa Barbara, 2001.
[21] F. Rabitti and P. Zezula, "A Dynamic Signature Technique for Multimedia Databases," in Proc. of the 13th International Conference on Research and Development in Information Retrieval, September 1990, pp 193-210.
[22] D. Shasha and K. Zhang, "Fast algorithms for the unit cost editing distance between trees. J. Algorithms," 11, pages 581–621, 1990.
[23] K. Tai, "The Tree-to-Tree Correction Problem," Journal of the ACM, 26(3), pages 422–433, july 1979.
[24] K. C. Tarjan, "Data Structure and Network Algorithms," CBMS-NSF Regional Conference Series in Applied Mathematics, 1983.
[25] P. Tiberio and P. Zezula, "Selecting Signature files for Specific Applications," Information Processing and Management, vol. 29, no. 4, pp. 487-498, 1993
[26] Y. Wang, D. J. DeWitt, and Jin-Yi Cai, "X-Diff: A Fast Change Detection Algorithm for XML Documents," Submitted for Publication, University of Wisconsin, Madison, 2002.
[27] J. Wang, K. Zhang, and D. Shasha, "A System for Approximate Tree Matching," IEEE Transactions on Knowledge and Data Engineering, 6(4):559–571, 1994.

[28] K. Zhang and D. Shasha, "Simple Fast Algorithms for the Editing Distance between Trees and Related Problems", SIAM Journal of Computing, 18(6): 1245-1262, 1989.
[29] K. Zhang, R. Statman, and D. Shasha, "On the Editing Distance between Unordered Labeled Trees," Information Proceedings Letters 42, pages 133–139, 1992.
[30] K. Zhang, J. T. L. Wang, and D. Shasha. On the Editing Distance between Undirected Acyclic Graphs and Related Problems, in Proc. of the 6th Annual Symposium on Combinatorial Pattern Matching, pages 395–407, 1995.
[31] http://www.hpl.hp.co.uk/people/bwm/rdf/jena/
[32] Resource Description Framework (RDF) Model and Syntax Specification, http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/
[33] RDF Primer http://www.w3.org/TR/2002/WD-rdf-primer-20020319/
[34] RDF Vocabulary Description Language 1.0: RDF Schema
http://www.w3.org/TR/rdf-schema/
[35] URI, Uniform Resource Identifier, http://www.mutu-xml.org/xml-base/shared/KEY-URI.pdf

Real World Semantic Web Applications
V. Kashyap and L. Shklar (Eds.)
IOS Press, 2002

Verifying Constraints on Web Service Compositions

Zhengang Cheng, Munindar P. Singh, and Mladen A. Vouk
Department of Computer Science
North Carolina State University
Raleigh, NC 27695-7535, USA
{zcheng, mpsingh, vouk}@eos.ncsu.edu

Abstract. Current service description and composition approaches consider simplistic method invocation. They do not accommodate ongoing interactions between service providers and consumers, nor do they support descriptions of legal protocols of interactions among them. We propose richer representations which enable us to capture more of the semantics of services than current approaches. Further, we develop algorithms by which potential problems in service compositions can be detected when services are configured, thereby leading to superior execution of composed services.

1 Introduction

Web services enable application development and integration over the Web by supporting program-to-program interactions. Relevant standards include Web Services Description Language (WSDL), Universal Description, Discovery and Integration (UDDI), and Simple Object Access Protocol (SOAP) [6, 18, 5]. These are intended, respectively, for describing, discovering, and invoking services. While current approaches represent much progress, they carry the baggage of traditional distributed objects approaches. Services are integrated through method invocation without regard to any higher-level constraints.

Current Web service techniques are limited to services where each operation is independent. A common example is stock-quote lookup, where each stock query is unrelated to every other query. But when we move from simplistic information lookup to interactive information search or to e-commerce, especially for complex business-to-business settings, it becomes obvious that current techniques are inadequate. In particular, we face two main challenges.

- What are useful ways of structuring the composition of services? Method invocation is appropriate for closed systems, but services are inherently autonomous and often to be used in long-lived interactions. For example, a long-lived interaction occurs in e-commerce when you try to change an order because of some unexpected conditions or try to get a refund for a faulty product. Even short-lived settings involve protocols, e.g., checking if the service requester is authenticated and properly authorized before accepting its order.

- What are appropriate semantic constraints on how services may participate in different compositions? As observed above, it may be required to compose services so as to carry out certain kinds of interactions. However, if a given service does not support such interactions, then the composition can fail at run time in unexpected ways. For example,

if an e-commerce service does not allow orders to be changed after a certain time has elapsed, it may be unsafe to use this service in certain settings, even if it is superior in other respects.

Current approaches, based on traditional closed systems, fall short of handling the above situations. WSDL allows us to capture the various methods but does not support constraints among those methods. Either too many methods will always be enabled or too few. However, WSDL's functionality is required to specify the methods supported by a service. Recently, Web Services Flow Language (WSFL) was proposed to describe compositions of services in the form of a workflow [14]. WSFL specifications tell us how different services ought to be invoked, e.g., in terms of ordering and parallelism among them, but they don't tell us whether a particular service that is bound in the workflow will in fact deliver the right interactions. XLANG [16] fits in Microsoft's BizTalk Server architecture. It describes the behavior of a single Web service. XLANG as it stands today provides a notation similar in spirit to workflow languages.

Our contribution is in capturing deeper constraints on what services are willing to offer, capturing richer requirements for service composition, and comparing the two to decide if a particular service is appropriate for the intended composition. Roughly, we think of WSDL as providing an underlying layer for our work. We enhance WSFL to define a service composition language that provides the necessary inputs to our reasoning approach. XLANG specifications are envisioned to drive automated protocol engines that will ensure that the specified message flows are obtained. Further, the intended direction is to define message exchanges among Web services. In this expanded form, XLANG will relate to our approach by providing elements of a service composition language.

Although the work reported here is still in an early stage of development, we believe it addresses important real-world concerns in the future expansion of the semantic Web. In particular, for the semantic Web technologies to penetrate real-life enterprise and scientific applications, the semantic Web will need as strong a representation of actions and processes as of data.

The rest of this paper is organized as follows. Section 2 introduces our representations and Section 3 applies them for verifying service compositions. Section 4 shows how our approach applies to service-level verification. Section 5 concludes with a discussion of the important themes, some of the relevant literature, and directions for further research.

2 Representing Composition Constraints

Because of their autonomy and heterogeneity, services are naturally associated with *agents*. Agents are long-lived, persistent computations that can perceive, reason, act, and communicate [10]. Agents act with varying levels of autonomy depending on environmental constraints and their previous commitments. Each agent provides a service and interacts through the exchange of messages, which denote business documents. We propose the Agent Service Description Language (ASDL) to describe the external behavior of agent services. An ASDL specification describes the messages understood by a service along with an interaction protocol it follows. Like WSDL, ASDL can be published and accessed through a registry. We propose the Agent Service Composition Language (ASCL) to describe the composition of new services from existing services. The imported services represent an agent role with its

behavior described in its ASDL description. An ASCL specification describes the interaction with other agent services.

2.1 Agent Service Description Language

ASDL enables us to define the behavioral characteristics of a Web Service. A Web service that implements behavior to preserve its autonomy is considered a Web agent service. In essence, ASDL is an behavioral extension to WSDL. It describes the constraints of service invocation to capture the external visible relationship between the operations.

The external behavior of the agent is demonstrated by its interaction with other agents. Such interactions occur through message exchanges. The internal implementation and reasoning logic is governed by the autonomy of the agent. For simplicity, we describe agent behavior via a finite state machine that models the allowed operation sequences.

The behavior of a service provider describes the allowed invocation sequence of operations. Invocations of this service must satisfy this sequence. We describe the agent behavior through a set of states and transitions between the states.

The state of an agent is used to maintain semantic constraints on actions. For example, a seller may require a buyer to log in before ordering some thing. Thus we can describe legal sequences of operation invocation. An operation state has a name for referencing and the operation it represents. For example, we can represent a state for the "order" operation defined in WSDL as follows.

<state name="order" operation="order"/>

ASDL contains states for all operations defined in WSDL with the name and operation attributes same as the operation name. We allow empty states which only have a name, but with no operation defined. They can be used to denote semantic states such as "ordered."

An interaction can proceed from one operation to another only if allowed by the modeled transitions. Each transition has a source operation and a destination operation. It may associate a condition to specify when the transition can occur. For example, the following specifies that the customer must successfully "Login" before "Ordering" any product. (The gating condition "expr" can involve terms from the messages exchanged in the protocol.)

<transition source="Login" target="Order" condition="expr"/ >

If there are no other transitions into "order," the customer must log in before ordering.

2.2 An E-Commerce Example

Figure 1 describes the behavior of an agent who requires login first, then enables querying and ordering products, leading to checkout and payment, and finally shipping.

We describe the seller's behavior with a behavioral extension of WSDL.

```
<behavior name="seller">
  <states>
    <state name="start" operation=""/>
    <state name="OQuery" operation="Query"/>
    <state name="Ordered" operation=""/>
  < /states>
  <transitions>
    <transition source="Start" target="Register" condition="NULL"/>
```

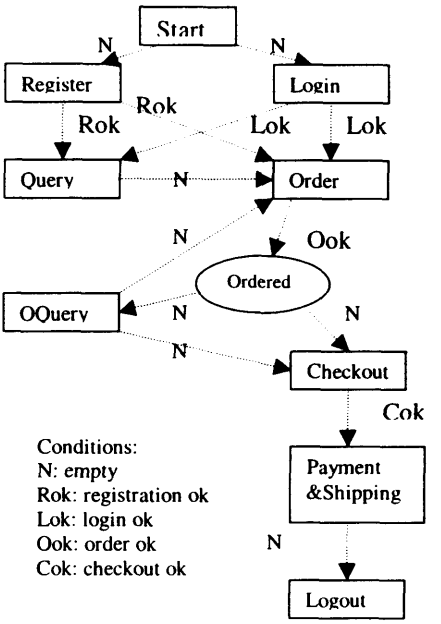


Figure 1: Behavior of a seller agent

```
<transition source="Start" target="Login" condition="NULL"/>
<transition source="Register" target="Query" condition="Rok"/>
<transition source="Register" target="Order" condition="Rok"/>
<transition source="Login" target="Query" condition="Lok"/>
<transition source="Login" target="Order" condition="Lok"/>
<transition source="Query" target="Order" condition="NULL"/>
<transition source="Order" target="Ordered" condition="Rok"/>
<transition source="Ordered" target="OQuery" condition="NULL"/>
<transition source="Ordered" target="Checkout" condition="NULL"/>
<transition source="OQuery" target="Checkout" condition="NULL"/>
<transition source="OQuery" target="Order" condition="NULL"/>
<transition source="Checkout" target="PaymentShip" condition="Cok"/>
<transition source="PaymentShip" target="logout" condition="PoK"/>
< /transitions>
< /behavior>
```

The following is a description of the transition conditions.

- NULL represents an empty condition.
- Rok, meaning registration OK, is a boolean expression based on the RegResult output message, and can be evaluated when the buyer receives the RegResult message.
- Lok, meaning Login OK, is a boolean expression based on the LoginResult output message.

- Ook, meaning Order OK, is a boolean expression based on the OrderResult output message. It indicates that the product is successfully ordered.
- Cok, meaning Checkout OK, indicates that an invoice is successfully send to the buyer.

The state *OQuery* represents that the buyer has ordered something and can check out at any time.

2.3 Agent Service Composition Language

Agent Service Composition Language (ASCL) specifications describe the logic of how a new service is composed from existing services. For reasons of space, we don't include details of ASCL syntax here. Suffice it to state that it enables services to be bound, and various flow primitives (sequencing, branching, and so on) used to specify desired compositions.

3 Verification of Service Composition

Using the above representations, we can determine whether service interaction protocols will be violated by the desired compositions. This can be done when services are bound—that is, at configuration rather than at run time.

We have developed some algorithms for reasoning based on the above representations. We lack the space to discuss these in detail. Briefly, these algorithms enable us to construct behavior state diagrams, find legal operations, and build an execution graph. We present the most interesting of these algorithms, which is for checking compliance of operation invocations.

To verify compliance at the operation level, we need the following data structures and functions.

- Build a service transition graph for each imported services.
- Find legal operations. Based on the service diagram, we can find the legal next states for a given current state.
- Remember history of invocation.

Given the above, following algorithm verifies a composition by traversing all possible execution paths.

1. Do a depth-first search on the resulted operation graph.
2. For each invocation of imported services, find the last state from the history, find the legal set of operation-states for the last state and current service from the corresponding service transition graph, check whether current operation is in the legal operation-state set. If not, there is an error. Otherwise add current operation to the history.
3. Continue until the whole operation graph is traversed.

Say a buyer finds the seller's service specification from the registry and would like to use the service to purchase a product from the seller. Figure 1 describes the seller's behavior in ASDL. It is up to the designer of the buyer agent on how to utilize the seller service. The verification algorithm is used for ensure that all possible execution paths in a service composition are legal. The following are some simplified examples to show how a service composition can be verified.

3.1 Sequential Composition

Suppose the buyer would like to implement the buy activity to buy products. (For brevity, unnecessary XML tags are not shown below.)

```
<consume role="seller" urlref="seller.xml"/>
  <activity name="buy">
    <sequence>
      <operation name="login" performedby="seller"/>
      <operation name="query" performedby="seller"/>
      <operation name="order" performedby="seller"/>
    < /sequence>
  < /activity>
```

Here the buyer invokes the login, query, and order operations of seller in sequence. Since login, query, and order are on the execution path, it is obvious that the above sequence is valid according to the seller's behavior.

3.2 Complex Composition

Now we consider the case where a buyer uses two seller services.

```
<consume role="S0" urlref="seller0.xml"/>
<consume role="S1" urlref="seller1.xml"/>
<activity name="buy">
  <operation name="register" performedby="S1"/>
  <fork condition="expr">
    <thread name="t1">
      <sequence>
        <operation name="login" performedby="S0"/>
        <if condition="expr">
          <sequence>
            <operation name="query" performedby="S1"/>
            <operation name="order" performedby="S0"/>
          < /sequence>
        < /if>
        <operation name="order" performedby="S0"/>
      < /sequence>
    < /thread>
    <thread name="t2">
      <sequence>
        <operation name="query" performedby="S0"/>
```

```
<switch variable="name">
  <case condition="expr2">
    <sequence>
      <operation name="order" performedby="S0"/>
      <operation name="query" performedby="S1"/>
    </sequence>
  </case>
  <case condition="expr3">
    <sequence>
      <operation name="query" performedby="S0"/>
      <operation name="checkout" performedby="S1"/>
    </sequence>
  </case>
</switch>
<operation name="order" performedby="S0"/>
</sequence>
</thread>
</fork>
</activity>
```

The above activity contains a *fork* construct. The threads in the *fork* construct are sequentialized. Based on the algorithm, we can build the execution graph as Figure 2. From Figure 2, we can find the following possible execution paths:

- Path 1: *S1:register, S0:login, S0:checkout, S0:query, S0:order, S0:query, S0:order*. Path 1 is not valid. The operation *S0:checkout*, is illegal since seller *S0* requires the buyer to *order* something before *checkout*.
- Path 2: *S1:register, S0:login, S1:query, S0:order, S0:checkout, S0:query, S0:order, S0:query, S0:order*. Path 2 is legal for both service provider *S0* and *S1*.
- Path 3: *S1:register, S0:login, S0:checkout, S0: query, S0: query, S1: checkout, S0:order*. Path 3 is illegal for the same reason as path 1.
- Path 4: *S1:register,S0:login, S1:query, S0:order, S0:checkout, S0: query, S0: query, S1: checkout, S0:order*. Path 4 is illegal, since the buyer has not ordered anything from seller S1, before trying to checkout.

4 Service Level Verification

The above discussion dealt with verifying service composition at the procedural level. Now we discuss verification at the service level. The service provider needs to verify that all possible invocation sequence of its exported services defined in its ASDL file are valid for all services it imported.

However, we have to note that it is not possible to enumerate all possible invocation sequences of a service. For instance, given a simple service S exposing two operations: a and b, the number of possible invocation sequences is infinite, e.g., a, b, ab, aa, bb, and so on. Here our verification is based on its exported service behavior defined in ASDL file. For example, the defined behavior is like $a->b$. We need to verify that the sequence a, b is valid.

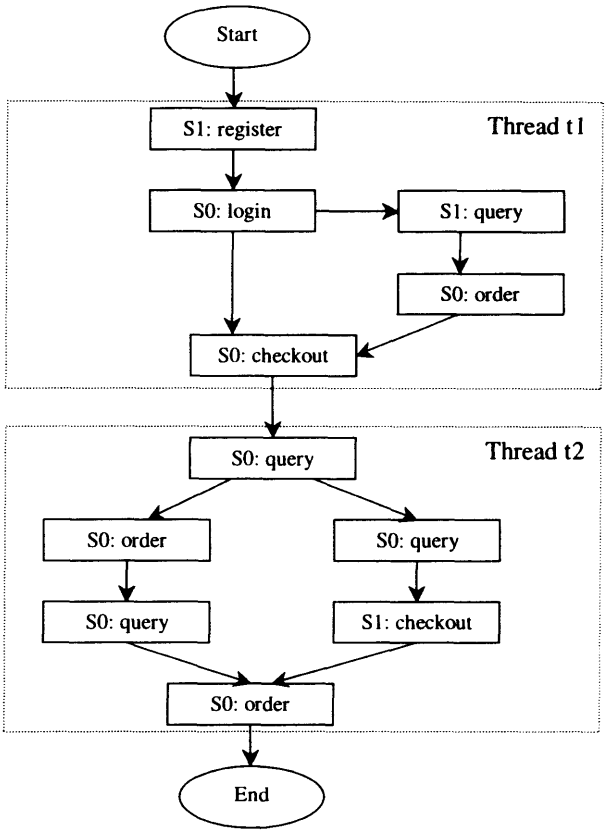


Figure 2: A composed service

Suppose there is a another operation c, which does not have an intrinsic relationship with a or b. We do not need to verify the validity of sequences involving c, since operation c is independent of operation a and b.

A service requester can be a final service consumer. It only use the service for its own purpose. Others are service integrators. They consume imported services and export a new service to the outside world. We need to find how verify at the service level.

For example, as in Figure 3, a service C is composed from imported services A and B. Service C has two exposed operations $c1$ and $c2$, and is then exposed as a new service to the outside world services like D and F. The behavior of service C is defined through a ASDL file, and published into a public registry. The ASDL definition can then be used to build new services manually or automatically. From the point view of Service C, it needs to ensure that invocation sequences following published service behavior in ASDL from outside like Service D, will not violate the restriction on the published behavior of services A and B. In service level verification, it needs to check whether all possible defined invocation sequences of operation $c1$ and $c2$ are valid.

Agents vary in their intelligence. Likewise, the services they provide can be simple or

complex. When a service consists only of behavior that can be captured by WSDL, the operations it supports are independent of each other. All invocation sequences of the stated methods are valid. For more complex services, some additional challenges must be addressed. Specifically, we consider the following complications during service level verification.

- Acyclic Directed Graph: The agent behavior can be described as a finite state machine. The verification can use algorithms similar to operation-level verification.
- Directed Graph with Loop: The agent behavior includes a loop. The verification algorithms can unravel the loop by using a modifier to limit the depth of the loop.
- Complex Directed Graph: The agent behavior can only be described by a complex graph. An example agent behavior could be like Figure 4, which consists of multiple loops in the definition. The service consists of four operations *m1, m2, m3, m4* with *S* and *E* designating the start and end state. How to verify a composed service against such complex services remains a challenge to be addressed.

At the service level, we can build a global execution graph by substituting the execution graph of each operation. For example, for a service *C* with exported operations *c1* and *c2* following the behavior *c1 –> c2*. The execution graph of operation *c1* is connected to the execution graph of operation *c2*, forming a global graph. Next, service compliance is checked against the execution graph.

This algorithm can handle acyclic directed graphs well, but it is not well-suited to complex graph that may include multiple loops as in Figure 4. In most practical situations, the service compositions are mostly coarse-grained with the invoked services doing substantial work. That is, the composition is relatively simple. The above algorithm should be enough for such situations.

4.1 Service Level Verification Example

Now we illustrate service level verification through the example of a travel service provided by a travel agent. The travel agent composes the services provided by hotel and airline agents. Its service is then exposed to customers. The customer may employ a customer agent to use the service.

We consider a simplified version of a travel agent. This only provides services such as query, purchase, and cancel. The travel agent requires the customer agent to login or register before purchase. A customer cannot cancel a ticket he has not yet bought. The airline and hotel agents require the travel agent to login, respectively, before booking a ticket or hotel room. They expose two operations *login* and *booking* and require the behavior that *login* should precede *booking*.

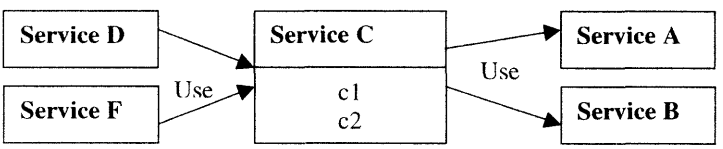


Figure 3: Service-level verification

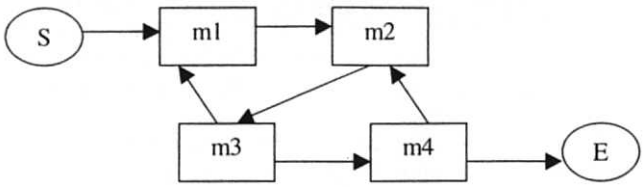


Figure 4: Complex service behavior of an agent

The travel agent exposes three operations, namely, *login*, *booking*, and *purchase*. Here, in operations *login* and *booking*, the travel agent will simply relay the request to the hotel and airlines corresponding to the *login* and *booking* operations. In the *purchase* operation, the customer will send both login credentials and a booking request to the travel agent; the travel agent will login to the airline and hotel agents, and book the ticket and room with them. The exposed behavior of these operations is that *login* should precede *booking*, while purchase is independent of the others. This scenario is illustrated in Figure 5. From the figure, it is easy to see that the service is valid for the composition.

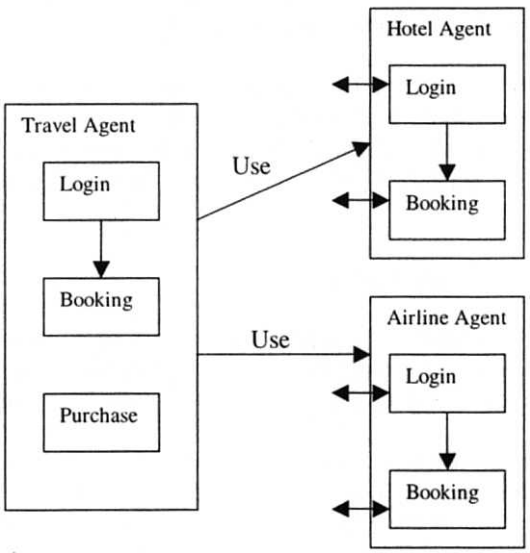


Figure 5: Travel agent scenario

Let us now consider an error situation. As shown in Figure 6, the travel agent exports *login* and *booking* operations. However, it tries to export a behavior wherein *booking* should precede *login*. In this case, the verification algorithm should detect the problem, because *login* should precede *booking* as required in its component hotel agent.

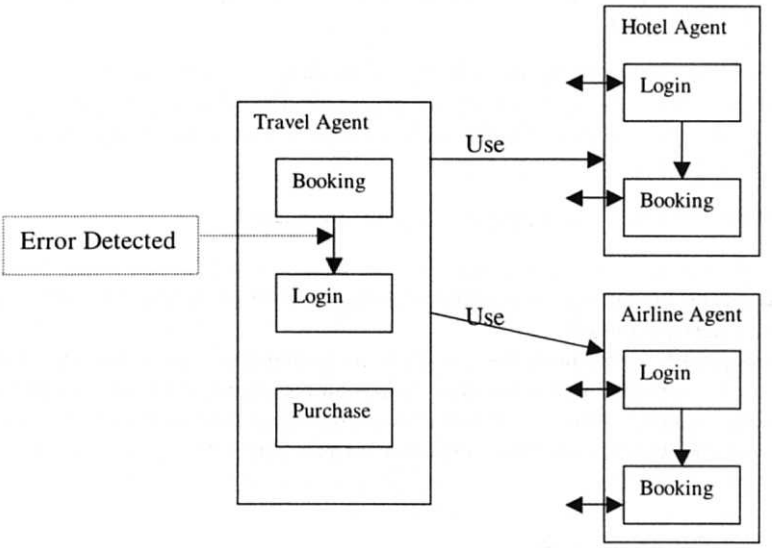


Figure 6: Service verification example: An error situation

5 Discussion

Although still in its infancy, the approach developed here seeks to facilitate the design and implementation of complex Web services as compositions of other Web services. Addressing this challenge is potentially of immense practical value. We believe it will be crucial to the expansion of semantic Web services into mainstream business process applications.

Our work treats Web service providers as autonomous agents, which can make independent decisions. ASDL exposes some behavior to the outside world. Our verification algorithms check the validity of a composed service, thereby detecting potential problems during the design phase of a composed service. We have developed a prototype tool to automatically check the validity of services.

5.1 Related Themes

Now we discuss some important topics concerning how our approach relates with service composition.

5.1.1 Automatic Service Composition

The above composition we considered is mainly at design time. For an autonomous agent, we might desire the agent to automatically come up with a plan of execution (a service composition based on other services). This remains a very difficult task. The DAML-S proposal provides a part of the foundation of automatic service composition, but much additional progress is necessary. DAML-S is a complex procedural language for web service composition. In particular, in order to achieve such a goal, the following problems must be addressed.

- Goal Definition: What states the composed service seeks to accomplish.

- Service Analysis: The agent should be able to analyze the goal and know what services might fulfill the goal.
- Service Selection: With the desired services known, the agent should be able the select individual services that can carry out the given task. This phase can be partially automated by using the classification of registries like UDDI and by some third party service rating services, e.g., [15].
- Execution Planning: To come up with a plan of execution.

Artificial intelligence tools such as Jess [7] may be applicable for reasoning, but defining a goal may still be too difficult in practice. Consequently, the user might be more inclined to compose the service himself.

Automatic Web service integration requires more complex functionality than SOAP, WSDL, and UDDI can provide. The functionality includes transactions, workflow, negotiation, management, and security. There are several efforts that aim at providing such functionality, for example, WSCL, WSFL, XLANG, BTP, and XAML. Entish [1] is also a relevant work in this direction.

5.1.2 Transactional Support

In real applications, transactional properties of services are important. It is reasonable that the service consumers or service providers may require two or more operations to be executed in as a transaction.

However, many business processes may need to run for a long time and it is not appropriate to run a whole process as a atomic transaction. Business processes can be modeled as long running processes, where the states and data can be stored, and be activated repeatedly over an extended period. To support such functionality, a messaging infrastructure, e.g., message queuing, should be in place between the service provider and consumer.

5.1.3 Error Handling

Error handling has not been adequately addressed in the context of composed services. In the real world, errors can occur anywhere between the service provider and service consumer. The service consumer should handle even possible networking errors. Service providers should expose the error messages in their ASDL. For example, the hotel agent might produce an insufficient funds error when attempting to charge the customer.

5.2 Literature

Besides Web services, the work described here touches upon extensive bodies of research on the semantic Web, workflow modeling, protocols, and agent-based techniques. We lack the space to review these in detail (but some were cited in the above discussion), but mention representative work here.

- Semantic Web. DARPA Agent Markup Language (DAML) [9] enables the creation of ontologies in the description of specific Web sites. DAML-S [2] is a Web service ontology from the semantic Web community [4]. DAML-S provides a core set of markup language

constructs for describing the properties and capabilities of Web services. Some emerging approaches add structure to service descriptions through the use of ontologies, e.g., [17].

- Workflow and process modeling. Klein & Bernstein develop a richer approach for describing and indexing services based on process models [12]. Verharen develops a contract specification language, CoLa, to specify transactions and contracts [19]. Verharen's approach captures obligations involving actions, but does not allow the obligations to be manipulated dynamically. This is a possible line of extension for the present work.

- Protocols. Barbuceanu and Fox [3] develop a language, COOL, for describing coordination among agents. Their approach is based on modeling conversations through FSMs, where the states denote the possible states a conversation can be in, and the transitions represent the flow of the conversation through message exchange. They try to handle exceptions through error recovery rules. HP's Conversation Definition Language [8] has similar goals to ASDL. CDL provides an XML schema for defining valid sequences of documents exchanged between Web services. Like ASDL, it uses conversations to model the externally visible interaction model of the Web service.

- Agent-based exception handling. Klein & Dellarocas exploit a knowledge base of generic exception detection, diagnosis, and resolution expertise [13]. Specialized agents are dedicated to exception handling. Our approach is complementary, since it applies at design time and does not require extensive intelligence.

5.3 Directions

This work opens up several interesting directions for research, some of which we are pursuing actively. On the practical side, we are working on our prototype to enhance its representational capabilities for services. On the theoretical side, it will be helpful to explicitly incorporate extended transactions in our models to capture richer constraints on service behavior.

6 Acknowledgements

This work was supported in part by the DOE SciDAC grant/contract DE-FC02-01ER25484 NSF grants CSS-9624425 and DST-0139037.

References

[1] Stanislaw Ambroszkiewicz and Tomasz Nowak. Agentspace as a middleware for service integration. In *Proceedings of Engineering Societies in the Agents World II*, pages 134–159, 2001.

[2] Anupriya Ankolekar, Mark Burstein, Jerry R. Hobbs, Ora Lassila, David L. Martin, Sheila A. McIlraith, Srini Narayanan, Massimo Paolucci, Terry Payne, Katia Sycara, and Honglei Zeng. DAML-S: Semantic markup for Web services. In *Proceedings of the International Semantic Web Working Symposium (SWWS)*, pages 411–430, July 2001.

[3] Mihai Barbuceanu and Mark S. Fox. COOL: A language for describing coordination in multi agent systems. In *Proceedings of the International Conference on Multiagent Systems*, pages 17–24, 1995.

[4] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic Web. *Scientific American*, 284(5):34–43, 2001.

[5] Don Box, David Ehnebuske, Gopal Kakivaya, Andrew Layman, Noah Mendelsohn, Henrik Frystyk Nielsen, Satish Thatte, and Dave Winer. Simple object access protocol (SOAP) 1.1, 2000. www.w3.org/TR/SOAP.

[6] Erik Christensen, Francisco Curbera, Greg Meredith, and Sanjiva Weerawarana. Web services description language (WSDL) 1.1, 2001. www.w3.org/TR/wsdl.

[7] Ernest J. Friedman-Hill. Jess, the Java expert system shell, 1997. herzberg.ca.sandia.gov/jess.

[8] Kannan Govindarajan, Alan Karp, Harumi Kuno, Dorothea Beringer, and Arindam Banerji. Conversation definitions: Defining interfaces of Web services. http://www.w3.org/2001/03/WSWS-popa/paper20.

[9] James Hendler and Deborah L. McGuinness. DARPA agent markup language. *IEEE Intelligent Systems*, 15(6):72–73, 2001.

[10] Michael N. Huhns and Munindar P. Singh. Agents and multiagent systems: Themes, approaches, and challenges. In *[111]*, chapter 1, pages 1–23. 1998.

[11] Michael N. Huhns and Munindar P. Singh, editors. *Readings in Agents*. Morgan Kaufmann, San Francisco, 1998.

[12] Mark Klein and Abraham Bernstein. Searching for services on the semantic Web using process ontologies. In *Proceedings of the International Semantic Web Working Symposium (SWWS)*, pages 431–446, July 2001.

[13] Mark Klein and Chrysanthos Dellarocas. Exception handling in agent systems. In *Proceedings of the 3rd International Conference on Autonomous Agents*, pages 62–68, Seattle, 1999.

[14] Frank Leymann. Web services flow language. TR WSFL 1.0, IBM Software Group, May 2001.

[15] E. Michael Maximilien and Munindar P. Singh. Reputation and endorsement for Web services. *ACM SIGEcom Exchanges*, 3(1):24–31, 2002.

[16] Satish Thatte. XLANG, Web services for business process design, 2001. www.gotdotnet.com/team/xml-wsspecs/xlang-c/default.htm.

[17] David Trastour, Claudio Bartolini, and Javier Gonzalez-Castillo. A semantic Web approach to service description for matchmaking of services. In *Proceedings of the International Semantic Web Working Symposium (SWWS)*, pages 447–462, July 2001.

[18] UDDI technical white paper, 2000. www.uddi.org/pubs/Iru-UDDI-Technical-White-Paper.pdf.

[19] Egon M. Verharen. *A Language-Action Perspective on the Design of Cooperative Information Agents*. Catholic University, Tilburg, Holland, 1997.

Real World Semantic Web Applications
V. Kashyap and L. Shklar (Eds.)
IOS Press, 2002

Semantic Web and Grid Computing

Carole GOBLE
University of Manchester, UK

David DE ROURE
University of Southampton, UK

Abstract. Grid computing involves the cooperative use of geographically distributed resources, traditionally forming a 'virtual supercomputer' for use in advanced science and engineering research. The field has now evolved to a broader definition involving flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions and resources. This is closely related to the Semantic Web vision. In this chapter we introduce grid computing and discuss its relationship to the Semantic Web, explaining how grid applications can and should be applications of the Semantic Web; this is illustrated by a case study drawn from the life sciences. We indicate how Semantic Web technologies can be applied to grid computing, we outline some e-Science projects using Semantic Web technologies and finally we suggest how the Semantic Web stands to benefit from grid computing.

1. Introduction

In the mid 1990s Foster and Kesselman proposed a distributed computing infrastructure for advanced science and engineering, dubbed 'The Grid' [1]. The name arose from an analogy with an electricity power grid: computing and data resources would be delivered over the Internet seamlessly, transparently and dynamically as and when needed, just like electricity. The Grid was distinguished from conventional distributed computing by a focus on large-scale resource sharing, innovative science-based applications and a high performance orientation. In recent years the focus has shifted away from the high performance aspect towards a definition of the 'Grid problem' as "flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and resources – what we refer to as virtual organizations." [2]

The Semantic Web Activity statement of the World Wide Web Consortium (W3C) describes the Semantic Web as "...an extension of the current Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation. It is the idea of having data on the Web defined and linked in a way that it can be used for more effective discovery, automation, integration, and reuse across various applications. The Web can reach its full potential if it becomes a place where data can be shared and processed by automated tools as well as by people." [3]

The Grid is frequently heralded as the next generation of the Internet. The Semantic Web is proposed as the (or at least a) future of the Web [4]. Although until very recently the communities were orthogonal; the visions are not, and neither should be the technologies. Grid computing applications can and should be seen as Semantic Web applications [5].

In this chapter we provide an overview of grid computing and discuss its relationship to the Semantic Web. We commence, in sections 2 and 3, with an introduction to the origins and evolution of grid computing. In section 4 we discuss the relationship between

the Grid and the Semantic Web visions, and then focus on a life sciences grid computing scenario in section 5. After a recap of Semantic Web technologies in section 6, we look in section 7 at the ways in which such a grid computing scenario can benefit from the Semantic Web. In section 8 we introduce some e-Science projects which are using Semantic Web technologies, and in the closing discussion of section 9 we suggest how the Semantic Web stands to gain from grid computing.

2. Origins of Grid Computing

The origins of the Grid lay in 'metacomputing' projects of the early 1990s, which set out to build virtual supercomputers using networked computer systems – hence the early emphasis on high performance applications. For example, the I-WAY project [6] was a means of unifying the resources of large US supercomputing centres, bringing together high performance computers and advanced visualization environments over seventeen sites. In contrast, the FAFNER (Factoring via Network-Enabled Recursion) project ran over networked workstations – described as 'world-wide distributed computing based on computationally enhanced Web servers' [7]. In both cases the goal was computational power and the challenge was finding effective and efficient techniques to utilise the networked computational resources, be they supercomputers or workstations.

Increasing the computational power by combining increasing numbers of geographically diverse systems raises issues of heterogeneity and scalability. These distributed computing infrastructures involve large numbers of resources – both computational and data – that are inevitably heterogeneous in nature and might also span numerous administrative domains. Scalability brings a number of challenges: the inevitability of failure of components, the significance of network latency so that it is necessary to exploit the locality of resources, and the increasing number of organisational boundaries, emphasising authentication and trust issues. Larger scale applications may also result from the composition of other applications, which increases the complexity of systems.

Rather than developing a series of 'vertical' grid applications, the vision of the Grid is an infrastructure which delivers computing and data resources seamlessly, transparently and dynamically as and when needed. This involves the development of middleware to provide a standard set of interfaces to the underlying resources, addressing the problems of heterogeneity. The Globus project [8], which has origins in I-WAY, has developed the best established grid middleware in current use. The Java-based UNICORE (UNiform Interface to COmputing REsources) project has similar goals [9].

The Grid priorities largely reflected the community that proposed it, that of High Energy Physics. Planned large-scale experiments, such as the Large Hadron Collider (LHC), capture and filter petabytes of data in a few seconds and complex simulations take months of computational processing. Subsequently, the benefits of grid computing have become apparent across a range of disciplines, such as life sciences.

Major exemplars of 'traditional' Grid include the following projects:

- The Information Power Grid (IPG) Project [10] is NASA's high performance computational grid that set out to establish a prototype production Grid environment. It has proven to be a significant Grid deployment, with a service-oriented approach to the architecture.
- The European DataGrid project [11] is setting up a computational and data-intensive Grid of resources for the analysis of data coming from scientific exploration such as LHC. It is led by CERN and funded by the European Union.

- The International Virtual-Data Grid Laboratory (iVDGL) for Data Intensive Science [12] has undertaken a very large-scale international deployment to serve physics and astronomy, building on the results of projects like DataGrid.
- TeraGrid aims to deploy 'the world's largest, fastest, most comprehensive, distributed infrastructure for open scientific research' [13]. It is based on Linux Clusters at four TeraGrid sites, with hundreds of terabytes of data storage and high-resolution visualisation environments, integrated over multi-gigabit networks.

The provision of computational resources in support of grid applications is supplemented by support for human interaction across the grid, known as Access Grid (AG) [14], which is designed to support group to group communication such as large-scale distributed meetings, collaborative work sessions, seminars, lectures, tutorials and training. Access Grid nodes are dedicated facilities that explicitly contain the high quality audio and video technology necessary to provide an effective user experience; they also provide a platform for the development of visualisation tools and collaborative work in distributed environments, with interfaces to grid software.

Given the nature of the Grid, there is clearly a role for a standardisation effort to facilitate interoperability of grid components and services, and this is provided by the Global Grid Forum (GGF). This is a community-initiated forum of individuals working on grid technologies, including researchers and practitioners. GGF focuses on the development and documentation of 'best practices', implementation guidelines and standards with 'an emphasis on rough consensus and running code', and has operated a series of international workshops [15].

3. Evolution of the Grid

Although motivated by a focus on high performance computing for High Energy Physics, the Grid approach is clearly applicable across a broad spectrum of scientific and engineering applications which stand to benefit from the integration of large scale networked resources. There is considerable investment in grid computing in the US, Europe and throughout the world. As further applications have been explored, the Grid has evolved in two dimensions both highly relevant for the Semantic Web: architecture and scope. These are explored in this section.

3.1 Architectural evolution: the service-based Grid

In order to engineer new grid applications it is desirable to be able to reuse existing components and information resources, and to assemble and co-ordinate these components in a flexible manner. The requirement for flexible, dynamic assembly of components is well researched in the software agents community [16] and is also addressed by the Web Services model, which has become established since the first 'Simple Object Access Protocol' (SOAP) standard was proposed in 1998.

The creation of Web Services standards is an industry-led initiative, with some of the emerging standards in various stages of progress through the W3C [17]. The established (sometimes de facto) standards, built on the Web languages XML and XML Schema as a transport mechanism, form layers to separate the concerns of transfer, description and discovery. Messages between services are encapsulated using SOAP; services are described using the Web Services Description Language (WSDL); services are registered for publication, finding and binding using Universal Description Discovery and Integration (UDDI).

The increasing acceptance of a service-oriented approach has led to a new service-oriented vision for the Grid: the Open Grid Services Architecture (OGSA) [18]. This brings the Grid in line with recent commercial and vendor approaches to loosely coupled middleware. Consequently, the e-Science and e-Commerce communities can benefit from each other, using industrial-strength tools and environments from major vendors.

However, the Grid's requirements mean that Grid Services considerably extend Web Services. Grid service configurations are highly dynamic and volatile, large and potentially long-lived. A consortium of services (databases, sensors and compute resources) undertaking a complex analysis may be switching between sensors and computers as they become available or cease to be available; hundreds of services could be orchestrated at any time; the analysis could be executed over months. Consequently, whereas Web Services are persistent (assumed to be available) and stateless, Grid Services are transient and stateful. Different priorities are also given to issues such as security, fault tolerance and performance. The influence of Grid Services has led, for example, to extensions in WSDL to deal with service instances and their state.

To achieve the flexible assembly of grid components and resources requires not just a service-oriented model but information about the functionality, availability and interfaces of the various components, and this information must have an agreed interpretation that can be processed by machine. Hence the emphasis is on service discovery through metadata descriptions, and service composition controlled and supported by metadata descriptions. Metadata has become key to achieving the Grid Services vision.

3.2 Scope evolution: the Information/Knowledge Grid

While the service-oriented view emerged to address the 'grid problem', another movement broadened the view of the Grid. Many e-Science activities (perhaps most) are more focused on the management and interoperation of heterogeneous information.

For example, the Life Sciences community is globally distributed and highly fragmented, so that different communities act autonomously producing tools and data repositories that are built as isolated and independent systems. Few centralised repositories exist except for critical resources. Most biological knowledge resides in a large number of modestly sized heterogeneous and distributed resources, including published biological literature (increasingly in electronic form) and specialised databases curated by a small number of experts. The complex questions and analyses posed by biologists cross the artificial boundaries set by these information-generating services.

We use "information generating services" rather than databases knowingly. Information is held in databases (and thus generated from them) but is also generated by instruments, sensors, people, computational analysis and so forth. The pressing need is to *weave together* information by finding it and linking it meaningfully. Astronomy, biodiversity, oceanography, geology are all characterised by the need to manage, share, find and link large quantities of diverse, distributed, heterogeneous and changeable information.

Keith Jeffery proposed organising conceptual services into three layers, illustrated in figure 1:

- A *data/computational grid* forms the fabric of the Grid to provide raw computing power, high speed bandwidth and associated data storage in a secure and auditable way. Diverse resources are represented as a single 'metacomputer' (virtual computer), so the way that computational resources are allocated, scheduled and executed, and the way that data is shipped between processing resources, is handled here.
- An *information grid* provides homogeneous access to heterogeneous distributed information by dealing with the way that all forms of information are represented,

stored, accessed, shared and maintained. This layer orchestrates data and applications to satisfy the request, including toolkits for composing workflows, accessing metadata, visualisation, data management, and instrumentation management. The Web, and other well-known and current middleware technologies are incorporated into one framework.

- A *knowledge grid* using knowledge based methodologies and technologies for responding to high-level questions and finding the appropriate processes to deliver answers in the required form. This last layer includes data mining, machine learning, simulations, ontologies, intelligent portals, workflow reasoning and Problem Solving Environments (PSEs) for supporting the way knowledge is acquired, used, retrieved, published and maintained. A knowledge grid should provide intelligent guidance for decision makers (from control room to strategic thinkers) and hypothesis generation.

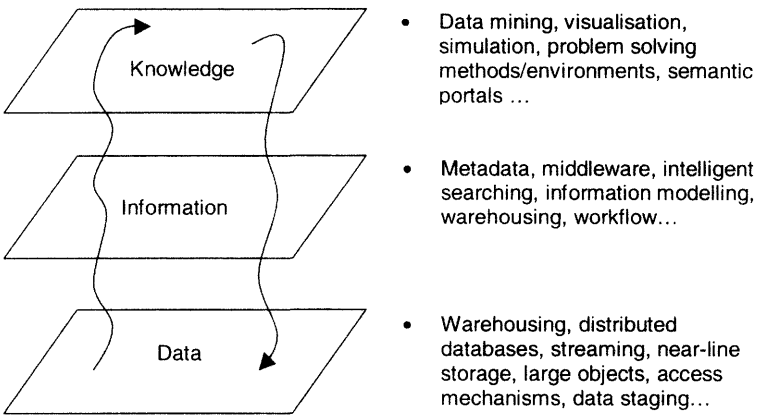


Figure 1. Three conceptual layers for the Grid (Jeffery)

Each layer represents a view for, or a context of, the previous layer. Multiple interpretations are possible at the junction between each layer. Each interpretation carries the context of whom or what is viewing the data, with what prior knowledge, when, why and how the data was obtained, how trustworthy it is etc. We can imagine a frame moving from bottom to top, so each will be re-interpreted as data for the next phase. Data could be measurements, the information a collection of experimental results and the knowledge an understanding of the experiment's results or its application in subsequent problem solving.

The layered model has proved useful to promote an expansion of the kind of services a Grid should support, although it has caused some confusion. In the original proposal the Knowledge Grid was where knowledge is generated rather than held; the Information Grid is where the knowledge is encoded. This has led to others merging the Knowledge and Information Grid into one. Whatever the semiotic arguments, in this expansion of the Grid vision, metadata is clearly apparent as an essential means of filtering, finding, representing, recording, brokering, annotating and linking information. This information must be shared and must be computationally consumable.

4. Relationship between the Semantic Web and Grid Computing

We have suggested that grid applications can be seen as Semantic Web applications, a step towards the 'Semantic Grid' [5]. Figure 2, which is based on a diagram by Norman Paton, captures the relationship between the two visions. The traditional grid infrastructure extends the Web with computational facilities, while the Semantic Web extends it with richer semantics. Hence we suggest that the evolving Grid falls further up the 'richer semantics' axis, as indicated by the dotted line in the figure.

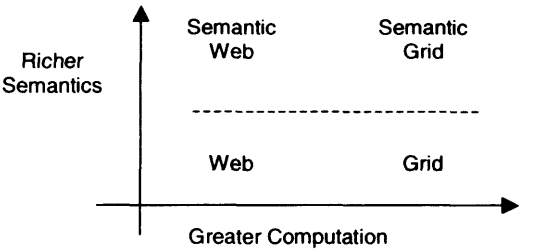


Figure 2. The Semantic Web and the Grid

Computationally accessible metadata is at the heart of the Semantic Web. The purpose of the Semantic Web is to describe a resource (anything with a URL) with what it is *about* and what it is *for*. Metadata turns out to be the fuel that powers engines that drive the Grid. Even before the Grid Service movement, metadata lay at the heart of the architecture diagrams of many grid projects. Figure 3 illustrates such an architecture.

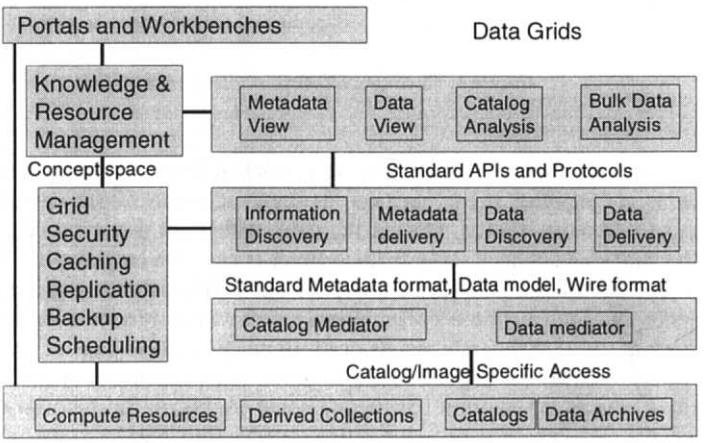


Figure 3. Example of Grid Architectures demonstrating the prevalence of metadata (NPACI)

The architectural/scope dimensions along which the Grid has evolved are orthogonal. We can use a similar duality when discussing the 'Semantic Grid' [5]. We can distinguish between a *Grid using semantics* in order to manage and execute its architectural

components (a Semantic Grid Services perspective) and a *Grid of semantics* based on knowledge generated by using the Grid – semantics as a means to an end and also as an end itself. The distinction is fuzzy of course and metadata will have a dual role. In this chapter we focus on the realisation of a Semantic Grid as a grid that uses Semantic Web technologies as appropriate, throughout the middleware and application.

To achieve the full richness of the e-Science vision – the 'high degree of easy-to-use and seamless automation and in which there are flexible collaborations and computations on a global scale' [5] – also requires the richness of the Semantic Web vision. This may include, for example, distributed inference capabilities, and working with inconsistent and changing data, metadata and ontologies. This is the territory above the dotted line in figure 2, and for practitioners it is important to distinguish between what is possible now and what may be possible in the future.

5. A Life e-Science Scenario

The Grid has been driven by e-Science. In this section we use an e-Science scenario from the Life Sciences to illustrate the full extent of the Grid vision, and draw comparisons with the Semantic Web.

Large-scale science is increasingly carried out through distributed global collaborations that will require access to very large data collections, very large scale computing resources and high performance visualisation. In practice, biology has already moved to large interdisciplinary teams distributed throughout the world working together on specific problems; e.g. the Human Genome Project.

The broad range of computational grid applications in this field includes protein-folding simulations, large scale sequence pattern matching, gene expression microarray mining and combinatorial chemistry. The computational power needed to model metabolic pathways or cells is huge. However, equally pressing is the fact that post-genomics and high throughput experimentation is promising to overwhelm the community with an avalanche of data that needs to be organised and harnessed. Advances in experimental techniques enable the rapid production of large volumes of data. The introduction of DNA microarray[1] technology is a good example of this.

Biological data is often complex, represented by different media, variable in quality, stored in many places, difficult to analyse, frequently changing and mostly comprised of incomplete data sets. Analysis methods to handle the different types of data are constantly and rapidly evolving. The questions asked of the data, and the computational analyses to ask them, are more complicated: multiple species rather than single species; whole genome rather than single gene; whole metabolic lifecycle rather than single biological process. Consequently, the traditional scientific experimental methods are supplemented with 'in silico experiments', for example, the prediction of genes and the metabolic pathways they encode from the genomic DNA of an organism.

Consider a biologist in a team examining the effect of neurotransmitters on circadian rhythms in Drosophila. Before conducting a microarray experiment she checks the literature and the laboratory's online lab books for whether any other similar experiment has taken place and if the data was already available. A sample is logged into a database and labelled. A set of parameters for the machine are inferred by past experiences of similar data for similar experiments, both by others and by the biologist from those used on previous runs (the microarray machine recognised the scientist from the log). The

[1] A microarray, in some ways resembling a computer chip, contains thousands of spots of known DNA samples. Hence in a single experiment, thousands of genetic elements can be studied simultaneously.

parameters are recorded with the output results, which are stored in her personal database alongside the image results. The results are immediately accessible by her from her office where she analyses them with a number of specialist statistical computations and a complex interactive time-series visualisation, both of which dynamically exploit a number of available computational resources to get better performance. The visualisation is examined collaboratively with a colleague on a remote site. Both scientists attach online personal notes to the results they share between themselves but are otherwise private.

Several proteins look interesting. In order to find the way that the functions of the clusters of proteins interrelate, three databases are linked together to find the proteins, group them into families and group and visualise them by their functions, as described by a controlled vocabulary held in a third database. Huge data repositories and complex queries will demand that the computational load is spread over the most available high performance machines. Other services will have to resolve the semantic differences between database schemas and content in different resources.

Papers, in free text, quoted in the database entries and extracted online from the Medline digital library reveal that, in certain circumstances, it could control genes related to the gene of interest. The system recommends other scientists who have published work or experiments that are related. From this she discovers the gene is a transcription factor. Valuable information may have to be extracted from free format text fields of flat files. These services require additional program logic to resolve problems associated with semantic heterogeneity of life science data sets.

The system inspects the biologist's laboratory's various 'transcriptome' databases, and discovers that genes that were co-regulated with the original gene also share a target site. This discovery activity requires a number of specialised services to interact with one another, some of which are data processing while others are computationally expensive. This information is added to a public database with a link to the workflow of database interrogations and analysis tools that lead to the discovery, including versions of databases, parameter settings, versions of the algorithms and the lab that made the discovery.

Suppose the geneticist wants to repeat the discovery process for all other over-expressed genes in the microarray data set. This is a data pipelining activity used to prove hypotheses, and requires the same chain of services to be applied to different data sets. If the scientist wants to express the methodology for this process in published work, she will want to preserve as much information about how services were used so her colleagues can replicate the activity. If the geneticist is working in a pharmaceutical company and the research proves commercially valuable, the company may want the ability to trace which proprietary was were used to support a conclusion. Both of these scenarios require the Grid to have an aspect of provenance, in order to track state information about various stages of work in a research activity.

Other scientists with appropriate access rights to this database who have run an analysis that included the gene in the last month are automatically notified with this new information. Another scientist incorporates the results into a simulation of a metabolic pathway they are running, using a problem-solving environment. The simulation is monitored by various colleagues around the world, who record both private and public observations. The simulation and its results are added to a public database, and trigger new simulations automatically.

Compare this scenario with that proposed as a vision for the Semantic Web by Tim Berners-Lee in the seminal Scientific American article [4], and those proposed by others [19, 20]. The similarities are striking. The Grid provides, and the scenario demands, a more comprehensive infrastructure, encompassing computational processes, security, authentication, accounting and so forth. However, a Grid application can be thought of as a Semantic Web application.

6. A Refresher on the Semantic Web Technologies

With the above scenario in mind, here we recap the vision of the Semantic Web. It is to evolve the web into one where information and services are understandable and useable by computers as well as humans. Automated processing of web content requires explicit machine-processable semantics associated with those web resources.

To realise this vision is in some ways mundane and in others daunting, depending on your ambition. As McBride points out [21], simple metadata and simple queries give a small but not insignificant improvement in information integration. Others have more ambitious ideas of an environment where software agents are able to discover, interrogate and interoperate resources dynamically, building and disbanding virtual problem solving environments [4], discovering new facts, and performing sophisticated tasks on behalf of humans. The key point is to move from a web where semantics are embedded in hard-wired applications to one where semantics are explicit and available for automated inference.

The core technologies proposed for the Semantic Web have their roots in distributed systems and information management:

- unique identity of resources by a URI and namespace scheme;
- annotation of resources with metadata for subsequent querying or manipulation;
- shared ontologies to supply the terms used by metadata in order that the applications or people that use it share a common language and a common understanding of what the terms mean (their semantics);
- inference over the metadata and ontologies such that unasserted facts or knowledge are inferred.

The minimal components needed for this vision include annotation mechanisms, repositories for annotations and ontologies with associated query and lifecycle management, and inference engines that are resilient, reliable and perform well. Then we need the tools to acquire metadata and ontologies (manually and automatically), describe resources with metadata and express metadata using ontologies, and for versioning, update, security, view management and so on. Such an environment forms a sandpit for search engines, information brokers and ultimately the 'intelligent' agents referred to by [4], that tend it, harvest it and enable it to blossom.

The 'layer cake' model of Figure 4 depicts Tim Berners-Lee's Semantic Web vision as a set of technology layers. URIs and Unicode provide standard ways to define references to entities and to exchange symbols. XML and XMLS enable machine-processable information. These provide the syntactic underpinnings of the Semantic Web.

The first layer of the Semantic Web itself is RDF which provides a means to represent the metadata that is needed to describe any kind resource, from a web page to a web service; "a foundation for processing metadata; it provides interoperability between applications that exchange machine-understandable information on the Web" [3]. RDF Schema defines a simple modelling language on top of RDF. Ontologies enable software agents to agree on the meaning of the content within web resources, and by providing definitions of key terms, allow them to interpret the meaning of metadata attached to resources. These definitions will be written using formal (logical) languages that facilitate automated reasoning. RDFS itself has expressive limitations; DAML+OIL [22], which is the basis of the OWL Web Ontology Language being designed by the W3C Web Ontology Working Group, provides the means to represent such ontologies, extending RDF Schema to define terminology in a restricted subset of first order logic. The automated reasoning supported by DAML+OIL/OWL, and further rule languages such as RuleML [23], infers new metadata and knowledge to classify services, discover alternative services or resources and ensure interoperability between services.

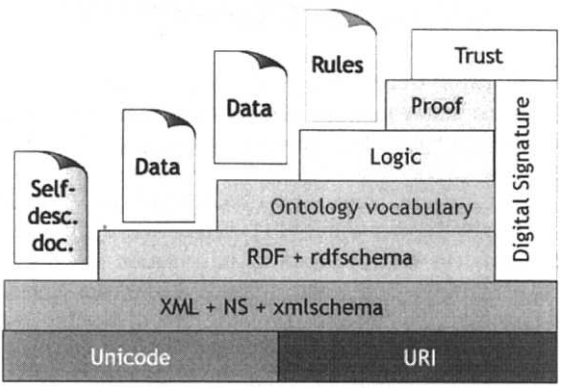


Figure 4. The Semantic Web Layer Cake (Berners-Lee, from XML2000 address)

The proof and trust layers are the least defined of the Semantic Web, but are intended to support provenance and confidence in the results. Proof is effectively exposing the reasoning used during inference combined with whether the facts and knowledge used to infer it are trustworthy. The confirmation that resources, metadata, knowledge and proofs are genuine is through digital signatures. These levels also relate to security and privacy. [24] suggests that these are applications where the other layers are languages.

7. Semantic Web Technologies in the Grid

Using the earlier scenario, we can explore some relationships with the Semantic Web vision and the opportunities for Semantic Web technologies.

7.1 Metadata based middleware

Metadata appears throughout the scenario and throughout a Grid application and all levels of the Grid. Consequently the metadata technologies developed for the Semantic Web are applicable and relevant. The following examples are generic to e-Science and hence pertain to the middleware rather than specific application domains:

- *Annotations* of results, workflows and database entries could be represented by RDF graphs using controlled vocabularies described in RDF Schema and DAML+OIL;
- *Personal notes* can be XML documents annotated with metadata or RDF graphs linked to results or experimental plans;
- Exporting *results* as RDF makes them available to be reasoned over;
- RDF graphs can be the "glue" that associates all the components (literature, notes, code, databases, intermediate results, sketches, images, workflows, the person doing the experiment, the lab they are in, the final paper) of an experiment, both in its *in silico* and 'at the bench' parts;
- The *provenance* trails that keep a record of how a collection of services were orchestrated so they can be replicated or replayed, or act as evidence, could for example be Web Service Flow Language (WSFL) [25] logs annotated by RDF-based metadata;
- Personalisation – the sculpting of resources into a personal "knowledge landscape" – is facilitated by descriptions relating to people and tasks.

Metadata is also exercised within the machinery of the grid computing infrastructure, for example:

- At the data/computation layer: classification of computational and data resources, performance metrics, job control, management of physical and logical resources;
- At the information layer: schema integration, workflow descriptions, provenance trails;
- At the knowledge layer: problem solving selection, intelligent portals;
- Governance of the Grid, for example access rights to databases, personal profiles and security groupings;
- Charging infrastructure, computational economy, support for negotiation; e.g. through auction model.

The metadata needs an agreed representation and, like any information, it has a lifecycle: it must be created, published, maintained and, ultimately, discarded. The creation of agreed metadata schema may be a community-led process. For example, bioinformatics content may be described using practices established in that community, while the grid infrastructure metadata schemas may be led by an organisation such as the Global Grid Forum. In the future, grid middleware could come with its own metadata schema and also support those of the application and users.

7.2 Dynamic marshalling and combining of resources

Complex questions posed by biologists require the fusion of evidence from different, independently developed and heterogeneous resources. The 400 or so public data repositories in active service in biology have different formats, interfaces, structures, coverage, etc. The Web and the Data Grid guarantee a certain level of interoperability in retrieving and accessing data. The next level of interoperability is not just making data available, but understanding what the data means so that it can be linked in appropriate and insightful ways, and providing automated support for this integration process [26]. Biologists are required to orchestrate resources in broadly two ways: (a) *Workflow orchestration*: Process flows, or workflows coordinating and chaining services using a systematic plan, are the manifestation of *in silico* experiments, allowing us to capture and explicitly represent the e-Scientist's experimental process; and (b) *Database integration*: dynamic distributed query processing, or the creation of integrated databases through virtual federations (e.g. TAMBIS [27]), portals or data warehouses (e.g. GIMS [28]).

Some schema belong to the application domain but others may be more generic (horizontal) to characterise grid computing resources. For example, coordinated distributed resource sharing applies to resources that are machines. Computationally intensive data analysis and predictive modelling can take advantage of spare resources available on machines connected to the Grid. Resources are discovered, allocated and disbanded dynamically and transparently to the user. Mediation between different Grid resource brokering models such as Unicore and Globus is a similar problem to mediating between two databases.

We can use Semantic Web technologies to:

- Represent the syntactic data types of Life Science objects using XML Schema data types, and use name spaces with URIs to uniquely identify instances of Life Science objects (the Life Science Identifer or LSID) [29, 30];
- Represent domain ontologies for the semantic mediation between database schema [26], an application's inputs and outputs, and workflow work items [31];

- Represent domain ontologies and rules for parameters of machines or algorithms to reason over allowed configurations;
- Use reasoning over execution plans, workflows and other combinations of services to ensure the semantic validity of the composition [32];
- Use RDF as a common data model for merging results drawn from different resources or instruments;
- Capture the structure of messages that are exchanged between components. This is an example where RDFS itself may appear inadequate (for example, it does not support cardinality constraints) but the design of RDF Schema can be informed by a knowledge of DAML+OIL.

There will not be just one ontology, one interpretation of results nor one set of metadata per data item. Brokering the diversity of interpretations is as much a Semantic Web vision.

7.3 The descriptive nature of the information

Partially as a result of a strong document publication ethos, knowledge in biology manifests itself in the literature and in elaborate metadata or "annotations" attached to raw data. Annotations are the accumulated knowledge attributed to a sequence, structure, protein, etc and are typically semi-structured texts. Many repositories and results are exported as flat files; the range of XML formats in Life Sciences is large and accelerating.

The premise is that a scientist will read and interpret the texts, but this makes automatic processing hard and is not sustainable given the huge amount of data becoming available. Where can the Semantic Web technologies contribute? This problem is the very problem the Semantic Web is intended to address.

The Life Sciences community has a familiarity and experience with descriptive metadata and the controlled vocabularies or ontologies required to manage it, for example the Gene Ontology [33]. Data is frequently integrated not at the schema level but at the content level: resources use a shared ontology or controlled vocabulary and link through the shared values, sometimes known as domain maps [34]. The GOBO community [35] recently recommended that the plethora of bioontologies coming on stream be encoded in DAML+OIL as a common exchange language and because the reasoning support is crucial when building large collaborative community ontologies.

7.4 The computational inaccessibility of information and applications

Many bioinformatics repositories and applications have simple call interfaces without APIs or query languages. Many have interactive "point and click" visual interfaces, which are good for people and bad for automated processing. Again, making the computationally inaccessible accessible lies at the heart of the Semantic Web. The service-oriented approach to the engineering of applications will address this. Components of legacy applications can be 'wrapped' as services and incorporated into the service description schemes.

7.5 Provenance, quality, trust, proof and the experimental process

Both the results, and the way they were obtained, are high value. What the data is like and where it came from is as important as the data itself. As data collections and analytical

applications evolve, keeping track of commensurate changes is difficult, so "change notification" becomes essential. Our scientist will need to rerun their experiment if data changes, or new knowledge questions the underlying premise of the analysis. In biology, data is replicated in secondary databases through annotation. This "viral migration" of information makes the tracking of data even harder. Mistakes or discredited information are propagated and difficult to eliminate.

A distributed environment on the scale of the Grid requires a number of core services built into its fabric to govern the whole scientific environment: ownership and watermarking (who owns the resource); provenance, quality, audit, versioning (where did the data come from and when); authentication, security and confidentiality (who can access the resource); personalisation and configuration (my lab book is special to me) and so on. These are clearly applications of the Proof, Trust and Digital Signatures of the Semantic Web.

7.6 The instability of science

Information and knowledge in science is heavily contextual and often opinionated. Contexts change and opinions disagree. New information may support or contradict current orthodoxy held by an individual, a team or a community, leading to a revision of beliefs. Managing, retrieving and reusing changing and evolving content is a challenge. Dealing with changing content that is also irregular, inconsistent, incomplete, contradictory, uncertain and imprecise makes the exploitation of these knowledge assets difficult. In particular, science is not "linear" – huge jumps are made in thinking, but the past cannot be wiped. It is essential to be able to recall a snapshot of the state of understanding at a point in time. This is why provenance is so important. All inferences *must* be exposed or else the scientist will not use them.

The expectation that there are multiple assertions on a resource and that inference engines over those assertions should deal with the "dirtiness" of the world is a bedrock of the Semantic Web vision. Not only does the content change, of course, but so do the ontologies and rules that we use to infer new information. When an ontology changes in line with new ideas, this does not wipe the old inferences that no longer hold (and how do we propagate those changes?). Somehow they must continue to co-exist and be accessible.

Event notification – when any entity in the Grid changes and fires further processes – is considered such an essential of e-Science that it is a core service in the Open Grid Service Architecture specification of Grid services. In our scenario, changes to database items are registered by our scientist to replay her experiment and by others who are affected by the results she publishes. Event notification schemes typically register the topics that should be monitored for change – these topics can be described using Semantic Web ontology and metadata technologies.

Scientific data has a very long lifespan. We do not throw away scientific knowledge and as a society we have a duty of guardianship. This means that the data in the Grid and the information that is being used to support the execution of the Grid persists. In addition to tending and harvesting it, we need to 'weed' the Semantic Web.

7.7 Knowledge

In the scenario, the biologist was advised of parameter settings based on information in the Grid. She was helped to choose appropriate experiments and resources, and to plan the execution of both her *in silico* and *in vitro* experiments. She used a problem-solving

environment for building and monitoring her metabolic pathway simulation. The recording, and sharing, of workflows helps improve experimental practice by avoiding unnecessary replication of in silico experiments (or in vitro experiments for that matter); it also assists in setting up equipment or computational processes in appropriate ways, and helps ensure that the conclusions drawn are fully justified by the techniques used. These are all applications of, or for, the Semantic Web – personalised agents or services [20], semantic portals onto services [36], recommender systems, intelligent searching, text mining for metadata [37] and so on.

The rhetoric of the Semantic Web often presents it as a global knowledge base. The metadata on resources are the facts; the ontologies the terms; the inference engines and rule systems the reasoning tools. Our scientist might well want to pose the question "what ATPase superfamily proteins are found in mouse?" and get the answers (a) P21958 (from the Swiss-Prot database she has permission to access); (b) InterPro is a pattern database and could tell you if you had permission and paid. (c) Attwood's lab expertise is in nucleotide binding proteins (ATPase superfamily proteins are a kind of nucleotide binding protein); (d) Smith published a new paper on this in Nature Genetics two weeks ago; (e) Jones in your lab already asked this question.

7.8 Collaborative science

Knowledge about who has published a relevant paper or asked a question is one mechanism for support of collaboration between scientists. In the scenario, the visualisation is examined collaboratively with a colleague on a remote site – this is a synchronous interaction between scientists, which is another form of collaboration that may itself be facilitated by the Grid. For example, a visualisation may be computationally intensive and make use of the high bandwidth network, especially if it is interactive and collaborative.

The Access Grid [14] can support this style of interaction, where dedicated or room-based facilities are required, or else desktop solutions are possible. Potentially the collaboration need not be office-to-office but could involve scientists working within experimental laboratories or in the field. Knowledge technologies can additionally be used for personalisation of information, tracking issues through a series of meetings, recording meetings and identifying communities of practice – see section 8.4 for an example project investigating some of these issues.

7.9 Semantic Grid Services

In the section 3 we explained that today's Grid applications are in a broad range of disciplines, are service-oriented, and require metadata for discovery and utilisation of resources. The service-oriented approach can be implemented with extensions to Web Services.

At the time of writing, the current state of describing Grid Services through semantics rather than just syntax via WSDL or simple classifications is as follows: "The service description is meant to capture both interface syntax, as well as semantics ...Semantics may be inferred through the names assigned the portType and serviceType elements...Concise semantics can be associated with each of these names in specification documents – and perhaps in the future through Semantic Web or other formal descriptions" [38]. This is a challenge and an opportunity for the Semantic Web community. Bringing together the Semantic Web and Web Services has already attracted attention [39,40,41,42]. Using

DAML+OIL to build service classifications more powerful than UDDI's simple hierarchies has been explored in the myGrid project [31] (see section 8.1).

The description of a service is essential for automated discovery and search, selection, (imprecise) matching, composition and interoperation, invocation, and execution monitoring. This choice depends on metadata concerned with function, cost, quality of service, geographical location, and the original publisher. Classification of services based on the functionality they provide has been widely adopted by diverse communities as an efficient way of finding suitable services. The Universal Description, Discovery, and Integration specification (UDDI) supports web service discovery by using a service classification. In Biology, the EMBOSS suite of bioinformatics applications and repositories has a coarse classification of the 200 or so tools it contains, and free text documentation for each tool; ISYS [43] and BioMOBY [29] use taxonomies for classifying services. In the pre-Services Grid, The Metadata Directory Service (MDS) [44] and Metadata Catalog (MCAT) [45] resource directory frameworks defined the properties that can be used to query a resource.

Reasoning has a role to play, not just in the creation of the ontologies used to classify services but also in the matching of services. In Condor, a structural matching mechanism was used to choose computational resources [46]. The semantic matching possible through reasoning in languages such as DAML+OIL has been explored in Matchmaker and myGrid [31]. In an architecture where the services are highly volatile, and configurations of services are constantly being disbanded and re-organised, knowing if one service is safely substitutable by another is an essential, not a luxury.

We mentioned earlier that Grid Services extend Web Services in three important ways: service instances, soft state and long-lived service orchestrations. How this will affect the way Semantic Web technologies can describe and discover Grid services is open for research.

8. Some examples of Grid Projects using Semantic Web Technologies

In 2001 the UK government launched a $180 million programme to develop and deploy Grid technology to support the challenges of 'e-Science' – the large scale science carried out through distributed global collaborations enabled by the Internet. The emphasis is on developing the grid infrastructure through projects in close collaboration with application scientists, and includes testbeds aimed at testing grid technology in a number of distinct science and engineering areas in collaboration with industry and commerce. The UK Grid vision pays particular attention to the processes by which Grid applications contribute to the creation and delivery of information and knowledge, as well as to the underlying computational resource sharing issues.

The timing of the UK e-Science programme has intercepted the second generation of service-based information-oriented Grids. Without obligations to the legacy of the earlier generation Grid technologies, many projects have been able to adopt a service-oriented approach from the outset, with due attention to information and knowledge aspects. In fact the emphasis is on the science, for which the Grid is an infrastructure, rather than on the infrastructure itself – hence some projects have adopted a more holistic view, which starts with the e-Scientist and the laboratory rather than the socket on the wall. Consequently, the full vision of e-Science is one where there is 'a high degree of easy-to-use and seamless automation and in which there are flexible collaborations and computations on a global scale' [5]. This requires more than just metadata: it requires the broader Semantic Web vision.

We now give examples of four of these e-Science projects. All four have adopted Semantic Web technologies and a service-based approach.

8.1 myGrid (www.mygrid.org.uk)

The myGrid project aims to deliver the middleware required for a personalised collaborative problem-solving environment. The focus is on data-intensive e-Biology, specifically post-genomic functional analysis, and the provision of a distributed environment that supports the in silico experimental process, though the middleware should be generic. The aim is that the e-Scientist be able to:

- Compose workflows, integrate and query databases and access to digital libraries through information extraction from texts.
- Find and adapt workflows and services developed by others, through accessing metadata based on ontologies.
- Store partial results in local data repositories, have their own view on public repositories.
- Be better informed as to the provenance and the currency of the tools and data directly relevant to their experimental space.

The Grid becomes egocentrically based around the Scientist – 'myGrid'. The ultimate goal is to improve both the quality of information in repositories and the way repositories are used.

myGrid uses ontologies to describe the bioinformatics services it publishes and orchestrates. A suite of ontologies expressed in DAML+OIL provides: (a) sufficiently elaborate service classifications to express the domain faithfully; (b) a vocabulary for expressing service descriptions and (c) a reasoning process to manage the coherency of the classifications and the descriptions when they are *created*, and the service discovery, matching and composition when they are *deployed*. The ontology extends the DAML-S service ontology [40]. Although the work is at an early stage, it shows that DAML+OIL provides an effective language to describe the functionality of service in such a demanding arena as biology [31].

myGrid is a good exemplar of a 'Semantic Grid' project, in particular because it makes use of a reasoning capability. Services are sought initially by their scientific metadata (domain dependent) and use the business metadata (domain independent) to choose between services of equivalent scientific functionality. Candidate services may not necessarily be an exact match, so the Fact system [22] is used to reason over the properties of the service descriptions expressed in DAML+OIL to infer close matches that are substitutable. Services can be discovered, matched and selected both *before* the workflow is executed or *dynamically* during its execution. Services generate data that could be the input to another service or could be stored in a repository, or both. In a workflow, we need to ensure that the type of the data (for example nucleotide sequence) matches the service's input type. The semantic *type* of the data must match: for example, a collection of enzymes is permissible as input to BLASTp as enzymes are a kind of protein and BLASTp takes sets of proteins as an input. To guide the user in choosing appropriate operations on their data and constraining which operation should sensibly follow which, it is important to have access to the semantic type of data concerned. Consequently, service descriptions cover the type of their inputs and outputs, and the type should be carried with the data for future use in new workflows. The service discovery interface of the myGrid Portal is illustrated in figure 5.

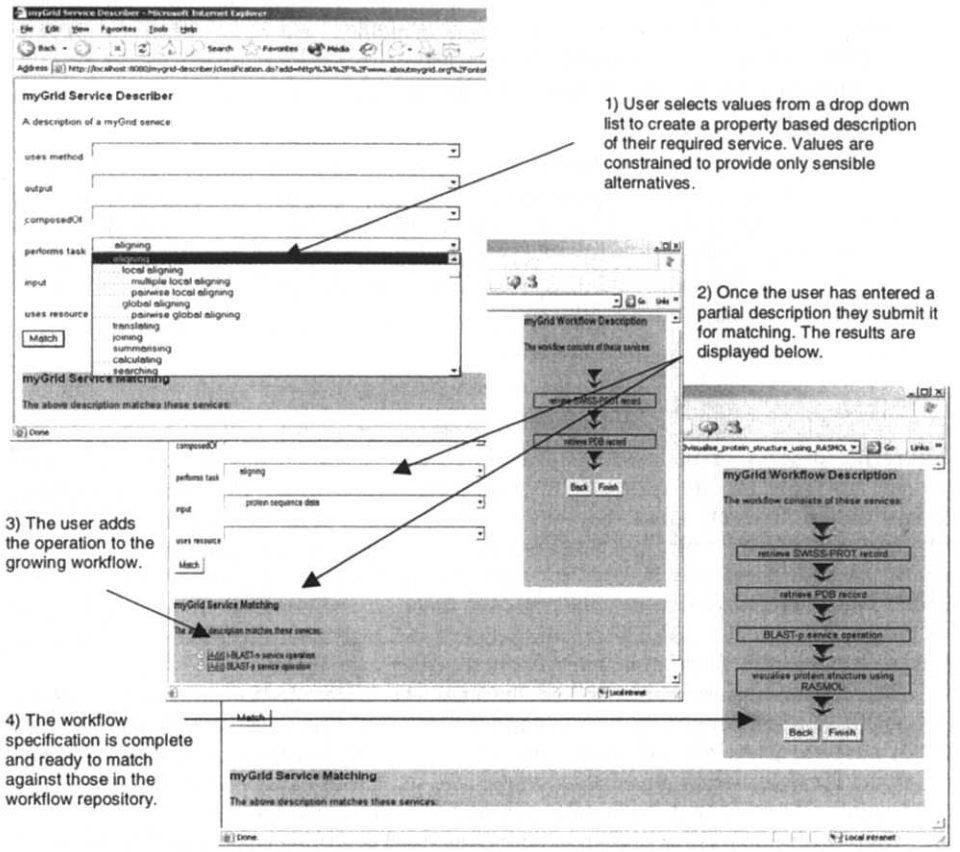


Figure 5. The service discovery interface of the myGrid Portal

8.2 Comb-e-Chem (www.combechem.org)

From an e-Science perspective, combinatorial chemistry has many similarities with the scenario in section 5. As its name suggests, combinatorial chemistry involves parallel synthesis methods that enable a large number of combinations of molecular units to be assembled rapidly – this is called a 'library'. In conjunction with this parallel synthesis there is parallel screening; e.g. for potential drug molecules. Each member of the library is tested against a target and those with the best response are selected for further study. When a significant response is found, the structure of that particular molecule is determined and used as the basis for further investigation to produce a potential drug molecule. As with the genetic databases, the size and growth rates of the databases of these molecules are dramatic.

The main objective of the Comb-*e*-Chem project is to develop an e-Science testbed that integrates existing structure and property data sources, and augments them in four ways within a grid-based resource- and knowledge-sharing infrastructure:

- by supporting new data collection, including process as well as product data, based on integration with electronic lab and e-logbook facilities;
- by integrating data generation on demand via grid-based quantum and simulation modelling to augment the experimental data;

- by developing interfaces that provide a unified view of these resources, with transparent access to data retrieval, on-line modelling, and design of experiments to populate new regions of scientific interest; and
- by providing shared, secure access to these resources in a collaborative e-science environment.

While some issues, such as provenance, are shared with myGrid, there is in Comb-e-Chem an emphasis on the experimental laboratory, on real time multimedia and collaborative aspects of e-Science, and on the design of experiments. The project has a strong automation theme, from robotics within the experimental laboratory through to automation of the e-Science workflows, and this is being addressed using the techniques of agent-based computing.

8.3 Geodise (www.geodise.org)

During the process of optimisation and design search, the modelling and analysis of engineering problems are exploited to yield improved designs. The engineer explores various design parameters that they wish to optimise and a measure of the quality of a particular design (the objective function) is computed using an appropriate model. A number of algorithms may be used to yield more information about the behaviour of a model, and to minimise/maximise the objective function, and hence improve the quality of the design. This process usually includes lengthy and repetitive calculations to obtain the value of the objective function with respect to the design variables.

The Geodise project aims to aid the engineer in the design process by making available a suite of design optimisation and search tools, Computational Fluid Dynamics (CFD) analysis packages integrated with distributed Grid-enabled computing and data resources. These resources are exposed to the user via a web-based portal. In addition, the user is guided through the design search process by an integrated knowledge base.

Ontologies serve as the conceptual backbone for knowledge sharing and management in the integrated architecture for knowledge services. The ontologies are represented in DAML+OIL and an ontology service provides a Java API giving full access to any DAML+OIL ontology available over the Internet. Annotation adds semantic content to documents or websites, thus facilitating information sharing, reuse and automatic machine processing. The OntoMat-Annotizer RDF annotation tool [47] is used to annotate workflows of optimization runs for particular design problems and then save them in a knowledge base. The semantically enriched archive can then be queried, indexed and reused later to guide future designs. A knowledge portal makes the knowledge available and accessible; provides tools for knowledge reuse and exchange; provides security infrastructure; manages knowledge resources; supports an online forum, maintains mailing lists and disseminates the latest advances of the domain.

8.4 CoAKTinG (www.aktors.org/cooakting)

The Access Grid [14], introduced in section 2, is a collection of resources that support human collaboration across the Grid, such as large-scale distributed meetings and training. The resources include multimedia display and interaction, notably through room-based videoconferencing (group-to-group), and interfaces to grid middleware and visualisation environments. It relates to the concept of a collaboratory [48], which is a distributed research centre in which scientists in several locations are able to work together.

During a meeting, there is live exchange of information, and this brings the information layer aspects to the fore. For example, events in one space can be communicated to other spaces to facilitate the meeting. At the simplest level, this might be moving through the agenda, slide transitions or remote camera control. These provide metadata, which is generated automatically by software and devices, and can be used to enrich the conference and stored for later use. New forms of information may need to be exchanged to handle the large scale of meetings, such as distributed polling and voting. Another source of live information is the notes taken by members of the meeting, including minutes and issue tracking, and the annotations that they make on existing documents. Again, these can be shared and stored to enrich the meeting. A feature of current collaboration technologies is that sub-discussions can be created easily and without intruding – these also provide enriched content.

The CoAKTinG project ('Collaborative Advanced Knowledge Technologies on the Grid') will provide tools to assist scientific collaboration by integrating intelligent meeting spaces, ontologically annotated media streams from online meetings, decision rationale and group memory capture, meeting facilitation, issue handling, planning and coordination support, constraint satisfaction, and instant messaging/presence. A scenario in which knowledge technologies are being applied to enhance collaboration is described in [49].

The combination of Semantic Web technologies with live information flows is highly relevant to grid computing and is based here on experience within the hypermedia context [50]. Metadata streams may be generated by people, by equipment or by programs (e.g. annotation, device settings, data processed in real-time) and we envisage 'metadata distribution networks'. Potentially CoAKTinG involves many ontologies, including those for the application domain, for the organisational context, for the meeting infrastructure, for devices which are capturing metadata and a constraint-based ontology for processes and products [51]. In contrast with some other projects, it requires real-time processing. For example, when someone enters the meeting, other participants can be advised immediately on how their communities of practice intersect.

9. Discussion and Conclusions

The Web was incubated by a scientific community: Physics. This community was a well-organised microcosm of the general community. It had definite and clearly articulated information dissemination needs and it had a group of smart people prepared to co-operate, and with the means and desires to do so. The state of play of the Grid today is reminiscent of the Web some years ago. At this time there is limited deployment, largely driven by enthusiasts within the scientific community (indeed, the High Energy Physics Community again), with emerging standards and a degree of commercial uptake. The same might also be said of the current state of the Semantic Web deployment, though it is not clear that the same drivers are in place as existed for Web and Grid.

Meanwhile, the Web itself has enjoyed massive deployment and continues to evolve; e.g. the shift from machine-to human communications (HTML) to machine-to-machine (XML), and the emergence of the Web Services paradigm. The requirements of one of the drivers, e-Commerce, are in line with those of e-Science. Thus the scene is set, and it is appealing to infer from these similarities that Grid and Semantic Web deployment will follow the same exponential model as the growth of the Web.

However, a typical grid application is not a typical web application. A grid application might involve large numbers of processes interacting in a coordinated fashion, while a typical Web transaction today still only involves a small number of hosts (e.g. server, cache, browser). Moreover, grid processes continually appear and disappear, while

web servers persist. Achieving the desired behaviour from a large scale distributed system involves technical challenges that the Web itself has not had to address, though Web Services take us towards a similar world. This is not just a data/computation layer issue; large scale integration of resources at information level is less well developed but no less challenging.

The Semantic Web requires a metadata-enabled Web. In the same way as the components of information systems have moved to support HTML and XML in the last few years, we now need them to take on board the support for creating and maintaining metadata. Unfortunately there are many obstacles to this. In particular, manual creation of metadata is problematic. People are not always in the best position to create it and they might not produce accurate metadata, through circumstance or error; even if accurate the metadata also needs to be useful, and there is 'more than one way to describe a cat'. With grid applications we have the requirement – and also the opportunity – to automate the management of *quality* metadata. Finally, creating ontologies is hard. Addressing the problems of creating and managing ontologies is the paramount.

So there are challenges ahead for both Grid and Semantic Web. Does this mean that the proposed marriage – the Semantic Grid – is ill-fated? We argue instead that the technologies are symbiotic and the partnership is essential for both to thrive.

We have shown that the visions of the Grid and the Semantic Web are related, and that Grid applications can be Semantic Web applications. Grid computing can gain immediately from the metadata technologies of the Semantic Web, informed by the OWL Web Ontology Language. The synergy is such that the Grid will also gain from the ongoing developments of the Semantic Web – for example, from the incorporation of inference technologies in the future – taking us towards the full Semantic Grid vision.

To achieve these benefits requires that the grid computing and applications community pay due attention to the Semantic Web. This applies to vertical projects, where the Semantic Web technologies can be applied within the application domain, and also to middleware developments, which can build in the Semantic Web infrastructure. There is a cost to taking on board new technologies, and here the benefits may not always be immediate. The Semantic Web community has a role to play in supporting the initial uptake, especially as many traditional Grid developers regard themselves as systems-oriented and the adoption of knowledge technologies seems a stretch. One barrier to adoption is confusion over what can be achieved now and what is best treated as 'wait and see'.

Why should the Semantic Web researchers be interested in the Grid? It is 'just an application' but in fact a very special one – perhaps even a 'killer app' – for several reasons:

- It is a very good example of the type of application envisaged for the Semantic Web. The essence of the Grid is the power provided by large scale integration of resources, and the scale and automation of the Grid necessitates the 'universally accessible platform that allows data to be shared and processed by automated tools as well as by people'.
- It is a real application: the emphasis is on deployment and on high performance, and is on a large scale and has established communities of users. Such applications are essential to the uptake of the Semantic Web.
- The Grid genuinely needs Semantic Web technologies. Even at the most basic level, Grid developers acknowledge that 'information islands' are being created and require an interoperability solution at information level such as provided by grid middleware at data/computation level.
- It will stress Semantic Web solutions, and it raises some specific grid-related issues which will provide a useful challenge. Solutions to these issues are unlikely to be

peculiar to grid computing – related issues will surely be evident in other Semantic Web applications in the fullness of time.
- It is self-contained, with a well-defined community who already work with common tools and standards.
- Aspects of the Semantic Web could be applications of grid computing, for example in search, data mining, translation and multimedia information retrieval.

The partnership between the Semantic Web and the Grid presents an exciting vision. Each partner has obstacles to its progress, but each stands to benefit from the other. To be successful, the partnership requires disjoint communities to come together. If they do, we can look forward to the 'next generation' of the Web: one with tremendous power to enable a new paradigm in science and engineering.

Acknowledgements

This work is supported by the Engineering and Physical Sciences Research Council and Department of Trade and Industry through the UK e-Science programme, in particular the myGrid e-Science pilot (GR/R67743), the Geodise e-Science pilot (GR/R67705), the Comb-*e*-Chem e-Science pilot (GR/R67729/01), the Semantic Grid report and the CoAKTinG project (GR/R85143/01) which is associated with the 'Advanced Knowledge Technologies' Interdisciplinary Research Collaboration (GR/N15764/01). The authors wish to thank all their colleagues who are working on the 'bridge' between the Grid and the Semantic Web. Nick Jennings and Nigel Shadbolt co-authored the Semantic Grid report for the UK e-Science programme, identifying a Semantic Grid research agenda, and we are grateful to all those who have supported this initiative within the UK including Tony Hey, Mark Baker, Malcolm Atkinson, Norman Paton, Omer Rana, Keith Jeffery, Tom Rodden and members of the Architecture Task Force, and to Hans-Georg Stork for his work on the European axis. We thank Ian Foster for his help and always being ready to discuss these ideas, both Ian and Carl Kesselman for bringing grid computing to the Semantic Web community through international conferences, and the Global Grid Forum for undertaking the Semantic Grid activity. Thanks especially to Jim Hendler for encouraging and supporting the discussion of grid computing within the Semantic Web community, and to Eric Miller and Henry Thompson from W3C for their contributions across the communities. Finally thanks to everyone currently building semantic grids: to Jeremy Frey, Simon Cox and Mike Surridge for exploring semantic grids through their grid computing projects, and to all our colleagues in the myGrid, Geodise, Comb-e-Chem, CoAKTinG, GRIA and AKT project teams.

References

[1] I. Foster and C. Kesselman (eds), "The Grid: Blueprint for a New Computing Infrastructure", Morgan Kaufmann, July 1998.
[2] I. Foster, C. Kesselman, S. Tuecke "The Anatomy of the Grid: Enabling Scalable Virtual Organizations", International Journal of Supercomputer Applications, 15(3), 2001.
[3] W3C Semantic Web Activity Statement, http://www.w3.org/2001/sw/Activity/
[4] Berners-Lee,T., Hendler,J. and Lassila ,O. "The Semantic Web", Scientific American, May 2001.
[5] D. De Roure, N.R. Jenning and N.R. Shadbolt. "Research Agenda for the Semantic Grid: A Future e-Science Infrastructure". Report UKeS-2002-02 from the UK National e-Science Centre, December 2001. Also see http://www.semanticgrid.org
[6] I. Foster, J. Geisler, W. Nickless, W. Smith, S. Tuecke "Software Infrastructure for the I-WAY High Performance Distributed Computing Experiment", in Proc. 5th IEEE Symposium on High Performance Distributed Computing. pp. 562-571, 1997.

[7] FAFNER, http://www.npac.syr.edu/factoring/
[8] I. Foster and C. Kesselman, "Globus: A Metacomputing Infrastructure Toolkit", International Journal of Supercomputer Applications, 11(2): 115-128, 1997.
[9] J. Almond and D. Snelling, "UNICORE: uniform access to supercomputing as an element of electronic commerce", Future Generation Computer Systems, 15(1999) 539-548, NH-Elsevier.
[10] NASA Information Power Grid, http://www.ipg.nasa.gov
[11] The DataGrid project, http://eu-datagrid.web.cern.ch
[12] iVDGL - International Virtual Data Grid Laboratory, http://www.ivdgl.org
[13] TeraGrid project, http://www.teragrid.org
[14] Access Grid, http://www.accessgrid.org
[15] Global Grid Forum, http://www.gridforum.org
[16] N. R. Jennings, "An agent-based approach for building complex software systems"", Comms. of the ACM, 44 (4) 35-41. 2001.
[17] W3C Web Services Activity, http://www.w3.org/2002/ws/
[18] I. Foster, C. Kesselman, J. Nick and S. Tuecke, "The Physiology of the Grid: Open Grid Services Architecture for Distributed Systems Integration", presented at GGF4, Feb. 2002. See http://www.globus.og/research/papers/ogsa.pdf
[19] J. Euzenat, "Research Challenges and Perspectives of the Semantic Web", European Commission - US National Science Foundation Strategic Research Workshop, Sophia Antipolis, France, October 2001.
[20] J. Hendler, "Agents and the Semantic Web", IEEE Intelligent Systems Journal, March/April 2001 (Vol. 16, No. 2), pp. 30-37.
[21] B. McBride, "Four Steps Towards the Widespread Adoption of a Semantic Web", in Proceedings of the First International Semantic Web Conference (ISWC 2002), Sardinia, Italy, June 9-12, 2002. LNCS 2342, pp 419-422.
[22] I. Horrocks, "DAML+OIL: a reason-able web ontology language", in Proceedings of EDBT 2002, March 2002.
[23] RuleML, http:// www.dfki.uni-kl.de/ruleml/
[24] P.F. Patel-Schneider, D. Fensel, "Layering the Semantic Web: Problems and Directions", in Proceedings of the First International Semantic Web Conference (ISWC 2002), Sardinia, Italy, June 9-12, 2002. LNCS 2342 pp 16-29.
[25] Web Services Flow Language (WSFL) Version 1.0, http://www-4.ibm.com/software/solutions/Webservices/pdf/WSFL.pdf
[26] C.A. Goble, "Supporting Web-based Biology with Ontologies", in Proceedings of the Third IEEE International Conference on Information Technology Applications in Biomedicine (ITAB00), Arlington, VA (November 2000), pp. 384–390.
[27] C.A. Goble, R. Stevens, G Ng, S Bechofer, N. Paton, P. Baker, M. Peim and A. Brass. "Transparent access to multiple bioinformatics information sources." IBM Systems Journal, Vol. 40, No. 2, pp 532-551. 2001.
[28] Paton, N.W., Khan, S.A., Hayes, A., Moussouni, F., Brass, A., Eilbeck, K., Goble, C.A., Hubbard, S. and Oliver, S.G., "Conceptual Modelling of Genomic Information", Bioinformatics, Vol 16, No 6, 548-558, 2000.
[29] BioMOBY http://www.biomoby.org
[30] Life Sciences Identifier (LSID), Interoperable Informatics Infrastructure Consortium (I3C), http://www.i3c.org
[31] C. Wroe, R. Stevens, C. Goble, A. Roberts, M. Greenwood, "A suite of DAML+OIL Ontologies to Describe Bioinformatics Web Services and Data", to appear in IJCAI Special issue on Bioinformatics Data and Data modelling.
[32] J. Cardoso and A. Sheth, "Semantic e-Workflow Composition", Technical Report, LSDIS Lab, Computer Science, University of Georgia, July 2002.
[33] M. Ashburner et al, "Gene Ontology: tool for the unification of biology". Nature Genetics 25: 25-29 (2000)
[34] B. Ludäscher, A. Gupta, and M.E. Martone, "Model-Based Mediation with Domain Maps", in 17th Intl. Conference on Data Engineering (ICDE), Heidelberg, Germany, IEEE Computer Society, April 2001.
[35] Gene Ontology Consortium, http://www.geneontology.org
[36] S. Staab, J. Angele, S. Decker, M. Erdmann, A. Hotho, A. Maedche, E. Studer,, and Y. Sure, "Semantic CommunityWeb Portals", in Proceedings of the 9th World Wide Web Conference (WWW9), Amsterdam, Netherlands, 2001.
[37] S. Staab, A. Mädche, F. Nack, S. Santini, L. Steels. "Emergent Semantics", IEEE Intelligent Systems, Trends & Controversies, 17(1), Jan/Feb 2002, pp. 78 –86.
[38] S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham and C. Kesselman, "Grid Service Specification Draft 3 7/17/2002", http://www.gridforum.org/ogsi-wg/

[39] D. Trastour, C. Bartolini and C. Preist, "Semantic Web Support for the Business-to-Business E-Commerce Lifecycle", in The Eleventh International World Wide Web Conference (WWW2002). pp: 89-98 2002.
[40] DAML Services Coalition, "DAML-S: Web Service Description for the Semantic Web", in The First International Semantic Web Conference (ISWC), June, 2002, pp 348-363.
[41] D. Fensel, C. Bussler, A. Maedche, "Semantic Web Enabled Web Services", in Proceedings of the First International Semantic Web Conference, Sardinia June 2002, Springer-Verlag LNCS 2342 pp:1-2.
[42] J. Peer, "Bringing together Semantic Web and Web Services", in Proceedings of the First International Semantic Web Conference, Sardinia June 2002, Springer-Verlag LNCS 2342 pp: 279-291
[43] ISYS, http://www.ncgr.org/isys/
[44] Globus Monitoring and Discovery Service, http://www.globus.org/mds/
[45] Meta Information Catalog, http://www.npaci.edu/DICE/SRB/mcat.html
[46] Condor, http://www.cs.wisc.edu/condor/
[47] S. Handschuh1 and S. Staab, "Authoring and Annotation of Web Pages in CREAM", in Proceedings of the Eleventh International World Wide Web Conference (WWW2002)
[48] V.G. Cerf et al., "National Collaboratories: Applying Information Technologies for Scientific Research", National Academy Press: Washington, D.C., 1993.
[49] S. Buckingham Shum, D. De Roure, M. Eisenstadt, N. Shadbolt and A. Tate, "CoAKTinG: Collaborative Advanced Knowledge Technologies in the Grid", in Proceedings of the Second Workshop on Advanced Collaborative Environments at the Eleventh IEEE Int. Symposium on High Performance Distributed Computing (HPDC-11), July 24-26, 2002, Edinburgh, Scotland.
[50] K.R. Page, D.G. Cruickshank and D. De Roure, "It's About Time: Link Streams as Continuous Metadata", in Proceedings of the Twelfth ACM Conference on Hypertext and Hypermedia (Hypertext '01) p.93-102. 2001.
[51] A. Tate, J. Levine, J. Dalton and A. Nixon, "Task Achieving Agents on the World Wide Web", in Creating the Semantic Web, Fensel, D., Hendler, J., Liebermann, H. and Wahlster, W. (eds.), 2002, MIT Press.

This page intentionally left blank

Semantic Web and Specialized Applications

This page intentionally left blank

Real World Semantic Web Applications
V. Kashyap and L. Shklar (Eds.)
IOS Press, 2002

Semantic Processing for Enhanced Access to Biomedical Knowledge

Thomas C. Rindflesch, Ph.D.
Alan R. Aronson, Ph.D.
National Library of Medicine, Bethesda, Maryland 20894

Abstract. The Semantic Knowledge Representation (SKR) project at the National Library of Medicine (NLM) develops programs that extract usable semantic information from biomedical text by building on resources currently available at NLM. Two programs in particular, MetaMap and SemRep, are being applied to a variety of problems in biomedical informatics. Both programs depend on the biomedical domain knowledge available in the Unified Medical Language System® (UMLS®). In representing concepts and relationships extracted from text, the semantic predications produced by these programs support a variety of applications in biomedical information management, including automatic indexing of MEDLINE® citations, concept-based query expansion, accurate identification of anatomical terminology and relationships in clinical records, and the mining of biomedical text for drug-disease relations and molecular biology information.

1. Introduction

An overwhelming amount of human knowledge is encoded in natural language texts (as opposed to databases), and the grand challenge in information technology is to provide reliable and effective access to this knowledge. For significant advances to be achieved, a richer representation of text is required than is currently available. The Semantic Knowledge Representation (SKR) project at the National Library of Medicine (NLM) develops programs that extract usable semantic information from biomedical text by building on resources currently available at NLM.

The Unified Medical Language System (UMLS) knowledge sources and the natural language processing (NLP) tools provided by the SPECIALIST system are especially relevant. The components of the UMLS provide structured representation of concepts and relationships in the biomedical domain. The Metathesaurus and the SPECIALIST Lexicon taken together represent names for concepts, while the Metathesaurus and the Semantic Network represent relationships between concepts. Natural language processing techniques are then called upon to provide a link between this domain knowledge and text.

Two programs in particular, MetaMap and SemRep, are being applied to a variety of problems in biomedical informatics. MetaMap maps noun phrases in free text to concepts in the UMLS Metathesaurus, while SemRep uses the Semantic Network to determine the relationship asserted between those concepts. As an example of the type of enhanced representation of text we are developing, (2) contains the semantic predication which represent some of the information contained in the text in (1).

(1) We used hemofiltration to treat a patient with digoxin overdose, which was complicated by refractory hyperkalemia.

(2) Digoxin overdose-OCCURS_IN-Patients
Hemofiltration-TREATS-Patients
Hemofiltration-TREATS-Digoxin overdose
Hyperkalemia-COMPLICATES-Digoxin overdose

Each of the predications in (2) is a proposition whose predicate (in upper case) is a relation from the UMLS Semantic Network. Each of the arguments is a concept from the UMLS Metathesaurus. The set of propositions in (2) considered as the semantic representation of (1) is not complete; however, it represents the major relationships and concepts contained in the text.

This approach to NLP, with its heavy dependence on the use of domain knowledge, follows in the tradition of semantics-oriented analysis. Classic work of this type includes that of Wilks [1], Schank [2], Riesbeck [3], and Hahn [4]. Maida and Shapiro [5][6] provide one viewpoint for representing the relationship between the assertions made in text and the interaction of entities expressed in a domain model of some possible world.

Bates and Weischedel [7] emphasize the importance of domain knowledge as a basis for significant progress in natural language processing effectiveness, while Saint-Dizier and Viegas [8] concentrate on lexical semantics in this regard. Sowa [9] discusses a number of important aspects of the interaction of domain knowledge and linguistic analysis.

There is currently a considerable amount of interest in natural language processing of biomedical text. Several approaches are being explored to provide reliable automatic analyses which can support practical applications. See, for example, Haug et al. [10], Hripcsak et al. [11], Friedman et al. [12], Rassinoux et al. [13], and Zweigenbaum et al. [14]. Hahn et al. [15] discuss the design of a semantic interpretation system that relies crucially on domain knowledge resources.

In the remainder of this paper we introduce the UMLS knowledge sources and then provide an overview of the NLP system we are developing in the SKR project. Finally, we describe some examples of applications (both completed and ongoing) that draw on the SKR system.

2. Unified Medical Language System (UMLS)

The UMLS is a compilation of more than 60 controlled vocabularies in the biomedical domain and is being constructed by the National Library of Medicine under an ongoing research initiative (Lindberg et al. [16], Humphreys et al. [17]) that supports applications in processing, retrieving, and managing biomedical text. The original hierarchical structure and relationships from each vocabulary are maintained, while synonymous terms across vocabularies are grouped into concepts. Information beyond that found in the constituent vocabularies is added by the UMLS editors, including semantic categories.

Component terminologies that provide broad coverage of the domain include Medical Subject Headings (MeSH®), Systematized Nomenclature of Medicine (SNOMED), International Statistical Classification of Diseases and Related Health Problems (ICD), Physicians' Current Procedural Terminology (CPT), and Clinical Terms Version 3 (Read Codes). Information focused in subdomains of medicine can be found in vocabularies such as Diagnostic and Statistical Manual of Mental Disorders (DSM), Classification of Nursing Diagnoses (NAN), WHO Adverse Drug Reaction Terminology (WHOART), and the University of Washington Digital Anatomist Symbolic Knowledge Base (UWDA). The Metathesaurus also contains a number of terminologies and vocabularies in languages other than English.

The UMLS is structured around three separate components: The Metathesaurus, the SPECIALIST Lexicon, and the Semantic Network. At the core is the Metathesaurus, which contains semantic information about more than 800,000 biomedical concepts, each of which has variant terms with synonymous meaning. Figure 1 shows some of the Metathesaurus information for Gaucher's Disease. Hierarchical structure from constituent vocabularies

Hierarchical Context:	Diseases Nutritional and Metabolic Diseases Metabolic Diseases Metabolism, Inborn Errors Lysosomal Storage Diseases Sphingolipidoses
Concept:	**Gaucher's Disease**
Synonyms:	GLUCOSYL CEREBROSIDE LIPIDOSIS Kerasin histiocytosis Cerebroside Lipidosis Syndrome ...
German:	Glukozerebrosidase-Mangelkrankheit Zerebrosid-Lipidspeicherkrankheit ...
Semantic Types:	**Disease or Syndrome** **Congenital Abnormality**

Figure 1. Part of the Metathesaurus concept for "Gaucher's Disease"

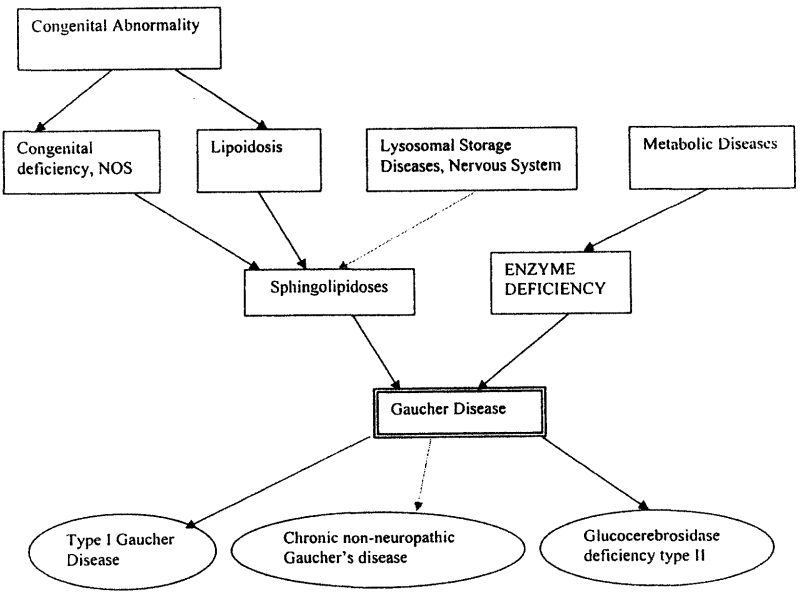


Figure 2. Some of the Metathesaurus concepts in relationships with "Gaucher's Disease"

forms the basis for relationships among concepts seen in the Metathesaurus. Using Gaucher's Disease as an example again, Figure 2 displays some of its relationships to other Metathesaurus concepts. English terms from the Metathesaurus are included in the SPECIALIST Lexicon, which contains more than 140,000 entries of general and medical terms and stipulates morphological and syntactic facts about English verbs, nouns, adjectives and adverbs (see Figure 3). Each concept in the Metathesaurus is also assigned a semantic category (or type), which appears in the Semantic Network, in which 134 semantic types interact with 54 relationships. Some of these semantic types and their relationships are shown in Figure 4.

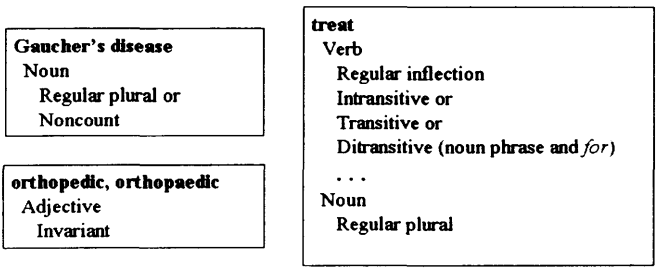


Figure 3. Examples of entries in the SPECIALIST Lexicon

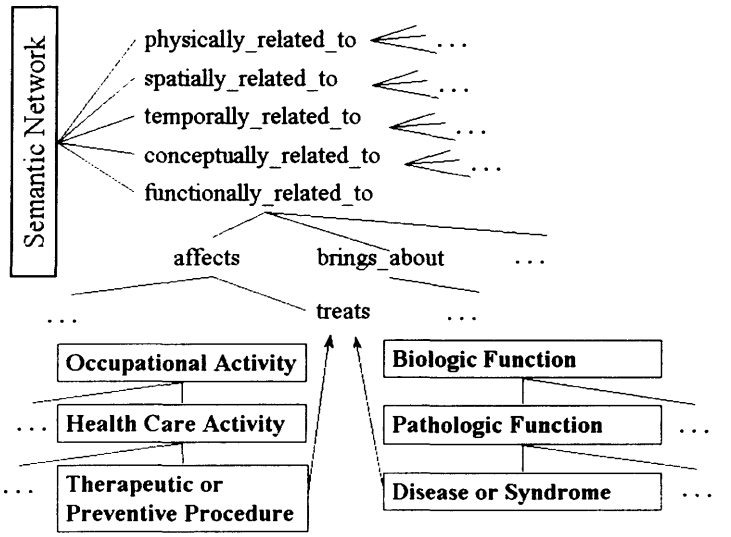


Figure 4. Part of the Semantic Network

3. Semantic Interpretation of Biomedical Text

Semantic processing in the SKR project draws on the resources being developed in the SPECIALIST NLP system (McCray et al. [18]; McCray [19]), which provides a framework for exploiting the resources of the UMLS in processing biomedical text. In addition to the Metathesaurus and Semantic Network, the SPECIALIST Lexicon and associated lexical variant programs (McCray, Srinivasan, and Browne [20]) as well as the Knowledge Source Server (McCray et al. [21]) support syntactic analysis and semantic interpretation of free text in the biomedical domain. At the core of the SKR effort are two programs, MetaMap (Aronson, Rindflesch, and Browne [22]; Rindflesch and Aronson [23]; Aronson [24]; Aronson [25]) and SemRep (Rindflesch and Aronson [26]; Rindflesch [27]; Rindflesch, Rajan, and Hunter [28]), which work in concert to provide the semantic representation given in example (2) above. An overview of NLP in the SPECIALIST system is given in Figure 5.

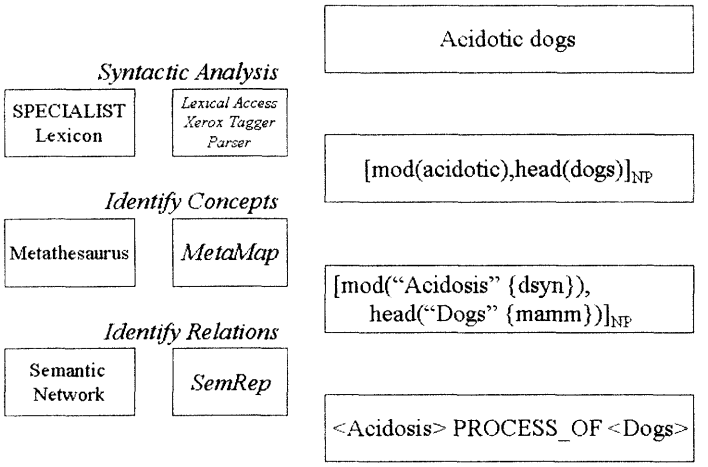


Figure 5. Overview of NLP in the SPECIALIST system

The SPECIALIST system begins analysis of biomedical text by consulting the Lexicon to determine syntactic information for each lexical entry in the input. A stochastic tagger (Cutting et al. [29]) is called to resolve part-of-speech ambiguities, and an underspecified syntactic analysis is produced as the basis for further processing. For example, input text *ablation of pituitary gland* is given the following analysis:

(3) [[head(ablation)] [prep(of), head(pituitary gland)]]

Although noun phrases are correctly identified, this analysis is underspecified in the sense that overall structure is not provided. That is, no commitment has been made to the exact relationship between the two constituent phrases produced. A further example of the characteristics of this type of analysis is given in (4), which is the underspecified analysis of the input text *pancreatic secretory trypsin inhibitor.*

(4) [[mod(pancreatic), mod(secretory), mod(trypsin), head(inhibitor)]]

In particular, note that, although the head of the noun phrase and its modifiers have been identified, no indication is given of the internal syntactic structure of such phrases. It is our hypothesis that this attenuated analysis is sufficient to serve as the basis for usable semantic interpretation.

The next step in processing calls MetaMap to get concepts from the Metathesaurus. This program takes advantage of syntactic analysis and considers each noun phrase individually as it proceeds. For example, it takes as input the underspecified syntactic analysis of *ablation of pituitary gland* and finds the following Metathesaurus concepts:

(5) Excision, NOS ('Therapeutic or Preventive Procedure', 'Research Activity')
Pituitary Gland ('Body Part, Organ, or Organ Component')

MetaMap accomplishes its task in four steps:

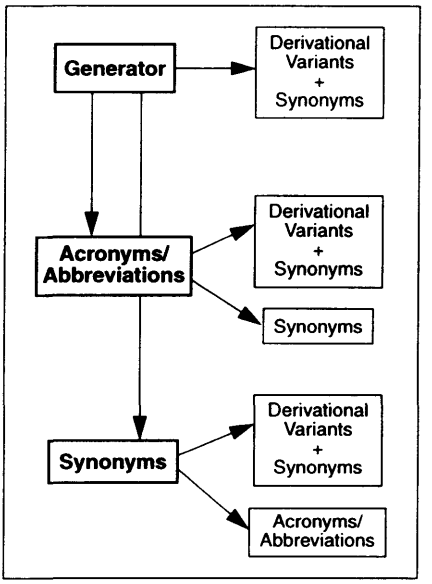


Figure 6. MetaMap variant generation (before inflections and spelling variants are computed)

- Variant generation: Each input word (or multi-word item, such as *wood alcohol*) generates a list of morphological variants, synonyms, and (optionally) acronyms/abbreviations plus meaningful combinations of these variants (Figure 6). For example, *aortic* and *arteria aorta* are variants of *aorta*;
- Candidate retrieval: Metathesaurus strings containing one or more of the input words are retrieved as candidates for a mapping. Some candidates for *arteriosclerosis* (in browse mode) are "Arteriosclerotic" and "Vascular Sclerosis";
- Candidate evaluation: Each candidate is evaluated for how closely it matches the input text according to a function with four components, centrality, variation, coverage and cohesiveness; and
- Mapping formation: Finally, candidates matching different parts of the input text are combined into a single mapping and re-evaluated to compute a total mapping score.

SemRep is called next and depends on both syntactic analysis and the Metathesaurus concepts provided by MetaMap. In addition, it consults the Semantic Network as part of the process of producing a final semantic interpretation. For example, in assigning an interpretation to *ablation of pituitary gland,* SemRep notes the syntactic analysis given for this input and then consults a rule which states that the preposition *of* corresponds to the Semantic Network relation LOCATION_OF, and further notes that one of the relationships in the Semantic Network with this predicate is

(6) Semantic Type 1: 'Body Part, Organ, or Organ Component'
Relation: LOCATION_OF
Semantic Type 2: 'Therapeutic or Preventive Procedure'

The MetaMap output for this input is then consulted, and it is noted that the Metathesaurus concept for the text phrase *ablation* is "Excision, NOS." The semantic type for this concept is 'Therapeutic or Preventive Procedure,' while the type for "Pituitary Gland" is 'Body Part, Organ, or Organ Component.' Since these semantic types match those found in the relationship indicated by the preposition *of* (LOCATION_OF) and since the relevant noun

phrases are allowable arguments of the preposition *of*, (7) is produced as the semantic interpretation for this phrase, where the corresponding Metathesaurus concepts are substituted for the semantic types in the Semantic Network relationship.

(7) Pituitary Gland-LOCATION_OF-Excision, NOS

A final example illustrates the application of MetaMap and SemRep to an entire MEDLINE citation (Figure 7). The output (Figure 8) provides a structured semantic overview of

Title: Long-term follow-up of a total articular resurfacing arthroplasty and a cup arthroplasty in Gaucher's disease.

Abstract: Patients with Gaucher's disease, a well-described lipid storage disorder with many systemic manifestations, often present to the orthopaedic surgeon with osteonecrosis of the femoral head. This can be a difficult orthopaedic problem because of the patient's young age at presentation and abnormal bone stock. Review of the literature leaves uncertainty as to the ideal treatment of femoral-head avascular necrosis in this disease. This article reports the long-term results of a cup arthroplasty and a total articular resurfacing arthroplasty procedure for bilateral hip involvement. In light of the less-than-satisfactory results of total hip arthroplasty for young patients with Gaucher's disease, resurfacing arthroplasty may warrant more serious consideration.

Figure 7. Title and abstract from a MEDLINE citation

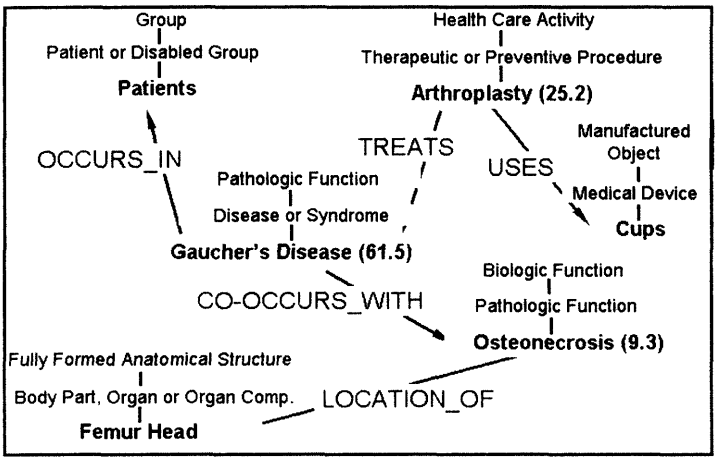


Figure 8. Semantic representation for the citation in Figure 7

the contents of this citation. Metathesaurus terms found by MetaMap in the text ("Patients," "Arthroplasty," "Cups," "Gaucher's Disease," "Osteonecrosis," and "Femur Head") represent the concepts central to this discourse. (Three concepts have been assigned an "aboutness" value indicating their high saliency.) UMLS semantic types (such as 'Therapeutic or Preventive Procedure' and 'Pathologic Function') associated with each concept provide

more general categorization. SemRep has computed Semantic Network relations between the concepts, which further enrich the representation of the content of this text.

4. Semantic Representation in Information Management Applications

SKR processing serves as the basis for a number of research projects that investigate the use of semantic knowledge representation for enhanced management of biomedical information. These projects are being conducted in collaboration with investigators at NLM and at other institutions.

MetaMap has been used for query expansion and indexing in research on concept-based information retrieval and in support of literature-based discovery systems. Most notably, MetaMap constitutes one of the core components of the Indexing Initiative system, which suggests automatically-generated indexing terms for MEDLINE citations.

SemRep has been applied to processing the research literature as well as clinical text. One project investigates data mining of drug-treatment relations from MEDLINE citations, while another identifies clinical findings in the literature on Parkinson's Disease. SemRep has also been applied to the task of identifying arterial branching relations in cardiac catheterization reports. Two further projects are aimed at extracting molecular biology information from text. The first of these addresses macromolecular binding relations, while the other is concerned with the interaction of genes, drugs, and cells in the research literature on molecular pharmacology for cancer treatment.

4.1 Information Retrieval Applications

Recent work (including that of Srinivasan [30][31][32][33]) has demonstrated the importance of query expansion based on retrieval feedback for improving retrieval effectiveness when applying statistically-based systems to MEDLINE citations. As an alternative method of query expansion, we have used MetaMap for associating Metathesaurus concepts with the original query. Our experiments show that this methodology compares favorably with retrieval feedback (Aronson and Rindflesch [34]).

MetaMap also served as the basis for research exploring full-text retrieval combined with techniques for hierarchical indexing (Wright, Grossetta Nardini, Aronson, and Rindflesch [35]). A subset of NLM's Health Services/Technology Assessment Text (HSTAT) database was processed with MetaMap, and the resulting Metathesaurus concepts were used in a hierarchical indexing method supporting information retrieval from full-text sources. Informal experiments suggested the value of this approach for improving results in both source and document selection when accessing large, multiple-source full-text document collections.

MetaMap was initially developed for improved retrieval of MEDLINE citations. The methodology was tested by applying this program to the queries and citations of the NLM Test Collection, replacing text with the Metathesaurus concepts discovered by MetaMap. Retrieval experiments using SMART were performed both on the unmodified test collection and on the MetaMapped version of the collection. The result was a 4% increase in average precision (Aronson, Rindflesch, and Browne [22]).

The Medical Text Indexer (MTI) system has been developed as part of the NLM Indexing Initiative (Aronson et al. [36]). In this project, several indexing methodologies are applied to the task of automatically indexing the biomedical literature, especially MEDLINE citations. The MTI system consists of three fundamental indexing methods:

- MetaMap Indexing (MMI), a linguistically rigorous method in which concepts found by MetaMap are ranked emphasizing either presence in the title or frequency of occurrence and the specificity of the concepts according to their depth in the MeSH hierarchy;
- Trigram Phrase Matching, a statistical method employing character trigrams to define a notion of phrases that are matched against Metathesaurus concepts; and
- PubMed® Related Citations, a variant of the "Related Articles" feature in PubMed to find articles that are similar to a citation of interest; selected MeSH headings from the most closely matching citations are the result of this indexing method.

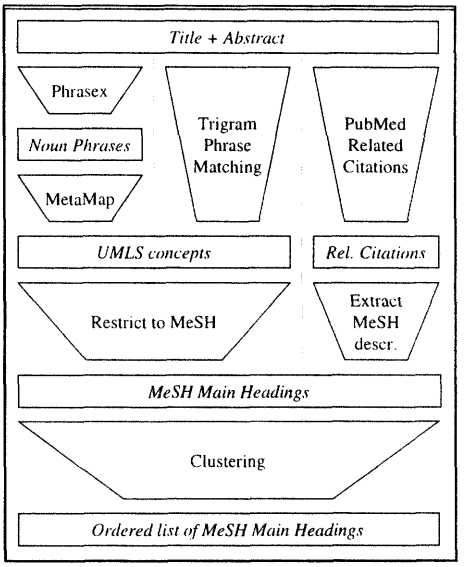


Figure 9. The Medical Text Indexer (MTI)

The first two basic methods produce UMLS Metathesaurus concepts that are then mapped to MeSH headings by the Restrict to MeSH method. This method uses relationships among Metathesaurus concepts to find the MeSH heading which is semantically closest to a given concept. The results of all the basic methods are combined using a Clustering method which generates a final ranking using several system parameters including weights for each of the basic methods.

The MTI system has been recently augmented with several postprocessing steps designed to increase the conformity of its recommendations to NLM indexing policy. MeSH headings which are never used for indexing are dropped; a choice is made between two headings, one of which is more specific than the other; modifications to the ranking of certain headings such as chemicals are performed; and several other similar changes are made. The result is a smaller number of recommendations which are generally more accurate than the original list.

The MTI system is beginning to be used both in computer-assisted and in fully automatic environments. NLM indexers can refer to the system's recommendations as they index MEDLINE citations using NLM's Document Control Management System (DCMS). MTI indexing is also being used in the NLM Gateway, a single retrieval system for accessing many of NLM's information resources, to index some collections which will not receive manual indexing. The collections include meeting abstracts on AIDS/HIV, health services research, and space life sciences.

4.2 Literature-Based Discovery

The DAD system developed by Weeber [37] is a concept-based literature discovery tool that enables biomedical researchers to explore domains which are not familiar to them but which may nonetheless provide information relevant to their research. Inspired by the work of Swanson [38][39], the DAD system relates knowledge about a disease C to a therapeutic substance A via a link B, typically a physiological process. So, for example, Swanson discovered that fish oil has therapeutic value for patients with Raynaud's Disease. The link here was the effect that fish oil has on platelet aggregation. The DAD system uses concepts found by MetaMap in MEDLINE citations to emulate the search for links between diseases and therapeutic substances, for example. The algorithm is focused by concentrating on concepts with appropriate semantic types at each stage of the search process. The combination of using concepts instead of words together with restriction by semantic types produces a more accurate as well as efficient methodology. Besides replicating Swanson's discoveries on Raynaud's Disease and another one on the therapeutic effect of magnesium on migraine headaches, the DAD system has been used to generate a hypothesis regarding the therapeutic application of an existing drug to a disease unrelated to the original intended use of the drug.

4.3 Text Mining

Text mining applications seek to extract knowledge from text that was not explicitly present in the source being mined. For example, MeSHmap (Srinivasan [40]) uses MeSH heading subheading combinations (the indexing terms assigned to MEDLINE abstracts) in order to provide semantic summaries of sets of documents retrieved by a user. Such summaries allow the user to explore relationships that were not overtly asserted in the input texts.

An extension of the MeSHmap technology (Srinivasan and Rindflesch [41]) uses SemRep in cooperation with MeSH indexing terms to provide increased confidence in identifying potentially interesting semantic relationships in large sets of MEDLINE citations. An example of the methodology is discussed on the basis of a set of citations having MeSH indexing terms in which a drug concept is modified by the subheading "therapeutic use" and a disease concept is modified by "drug therapy." For all such citations, the semantic interpretation of the title was obtained from SemRep. For example, the title in (8) was given the interpretational in (9). Two indexing terms assigned to the citation having this title are shown in (10).

(8) Pentoxifylline in cerebrovascular dementia

(9) Pentoxifylline-TREATS-Dementia

(10) Dementia, Multi-Infarct/diagnosis/*drug therapy/etiology
Pentoxifylline/administration & dosage/pharmacology/*therapeutic use

Relevant MeSH indexing terms combined with SemRep predications were extracted from more than 15,000 MEDLINE citations discussing drug therapies and diseases. Of the 7,332 drug-disease pairs identified, the five most frequent are shown in Table 1.

Further research is planned on the basis of concept pairs such as those in Table 1. When the entire list of extracted pairs is examined, it can be determined that certain drugs have been discussed in disease contexts of varying diversity. For example Srinivasan and Rindflesch [41] report that pyrithrioxin appears in a rather homogeneous context (largely Alzheimer disease and dementia), while pyridazines have been associated with an array of disorders, including congestive heart failure, depressive disorders, and the common cold. It is appeal-

ing to suggest that research such as this, in computing a "diversity index" for drugs and diseases encountered in the research literature, may provide useful information to the health care practitioner as well as the researcher. Intuitively, information about drugs with a high diversity index, such as pyridazines, may stimulate discovery regarding diseases and effective therapies.

Table 1. Most-Frequent Drug-Disease Pairs ("Occurrence" indicates the number of citations in which a pair appeared.)

Drug concept	Disease Concept	Occurrence
antihypertensive agents	hypertension	66
nifedipine	angina pectoris	43
calcium channel blockers	angina pectoris	41
atenolol	hypertension	39
propanolamines	hypertension	34

4.4 Processing Clinical Data

Focused use of MetaMap along with a number of rules describing the internal structure of findings formed the core of the FINDX program, which was developed to identify findings in clinical text and MEDLINE abstracts (Sneiderman, Rindflesch, and Aronson [42]). One of the conclusions drawn from this study is that the structure of findings in the research literature and in clinical text is essentially the same: an attribute of the patient under consideration is reported along with a value for that attribute. For example, FINDX identified findings such as *elevated liver function tests* in clinical text and findings such as *eight demented cases had absent neocortical neurofibrillary tangles* in the research literature. The program proceeds by first MetaMapping the input text to Metathesaurus concepts. A set of findings rules then looks for concepts having semantic types, such as 'Physiologic Function', or 'Laboratory Procedure' occurring in close syntactic proximity to values such as *absent, elevated, normal,* etc. FINDX was evaluated on a set of MEDLINE citations discussing the diagnosis of Parkinson's disease. A potential use for this application is filtering information retrieval results on the basis of the findings observed, perhaps in support of evidence-based medicine and clinical decision making.

Rindflesch, Bean, and Sneiderman [43] report on the use of MetaMap and SemRep for processing cardiac catheterization reports. Statements in the arteriography section of these reports describe characteristics of the arteries seen during the catheterization procedure, such as branching configurations and areas of stenosis. This project focused first on identifying the names for the coronary arteries and then on retrieving the branching relations that were asserted to obtain among the arteries observed. The focused nature of this application along with the complexity of the arterial terminology provided a useful context for development of the SKR methodology. Extensive reliance on UMLS domain knowledge contributed significantly to a highly accurate semantic analysis for these reports. For example, this processing identified the branching predications given in (12) for the text in (11), and the names for the coronary arteries in the text have been normalized to the corresponding Metathesaurus concepts.

(11) The left main gives off a left circumflex and left anterior descending branches.

(12) Anterior interventricular branch of left coronary artery-BRANCH_OF-Left coronary artery
Circumflex branch of left coronary artery-BRANCH_OF-Left coronary artery

The results from this project suggest the feasibility of extending this processing to a more comprehensive normalization of the semantic content of anatomically-oriented text. Such processing could support innovative applications of information management applied to both clinical text and the research literature.

4.5 Molecular Biology Applications

Several recent SKR projects involve the adaptation and extension of MetaMap and SemRep for extracting molecular biology information from the research literature. One such program, Arbiter, identifies macromolecular binding relationships in MEDLINE citations. Arbiter operates in two phases; the first (Rindflesch, Hunter, and Aronson [44]) identifies all binding entities mentioned in the input text and addresses such phenomena as molecules, genomic structures, cells and cell components, as well as topographic aspects of molecules, cells, and cell components. In order to identify these entities, Arbiter relies on MetaMap output and UMLS semantic types, such as 'Amino Acid, Peptide, or Protein', 'Nucleotide Sequence', 'Carbohydrate', 'Cell Component'. In addition, Arbiter calls on a small set of words that appear as heads of binding terms. These include words referring to various biomolecular phenomena (*box, chain, sequence, ligand,* and *motif*), molecular or cellular topography (*spike, cleft, groove, surface*), and general terms for bindable entities (*receptor, site, target*).

During the second phase of processing, Arbiter establishes binding relationships using the general SemRep machinery, focused on forms of the verb *bind* (Rindflesch, Rajan, and Hunter [28]). Binding relationships are semantically constrained to obtain only between the binding entities identified during the first phase of processing. As an example of Arbiter output, the predication in (14) was extracted from the text in (13).

(13) In support of this notion, we have found that aminohexose pyrimidine nucleoside antibiotics, which bind to the same region in the 28S rRNA that is the target site for anisomycin, are also potent activators of SAPK/JNK1.

(14) aminohexose pyrimidine nucleoside antibiotic-BINDS-28s rrna

Arbiter was evaluated on a test collection of 116 MEDLINE citations and was then run on 500,000 citations; some 350,000 binding predications were extracted and entered into a database for further analysis. Current research on Arbiter includes extending its application to protein-protein interactions in general (Sarkar and Rindflesch [45]) as a basis for investigating protein function similarities.

Molecular pharmacology for cancer therapy is characterized by the complexity involved in the interaction between drugs, genes, and cells. Genes affect drug activity and drugs affect gene expression; at the same time, both gene expression and drug activity vary across cell types. SKR and UMLS resources are being used as the basis for the development of a program called Edgar (Rindflesch, Tanabe, Weinstein, and Hunter [46]) that is being designed to address this complexity. The program is designed to first identify drugs, genes, and cells in text and then to determine interactions such as "over expresses" that involve these entities. Edgar identifies drugs, genes, and cells in MEDLINE citations using techniques similar to those used by Arbiter. Gene identification is enhanced by calling on several statistical and empirical methods devised by Tanabe and Wilbur ([47]). Although identification of semantic

relationships in this domain is still under development, SemRep underpins techniques being developed to extract, for example, the predications in (16) from the text in (15).

(15) Furthermore, RA treatment enhanced the transcriptional activity of a reporter construct containing the Sry/Sox consensus sequence in TC6 cells.

(16) RA-INCREASES_EXPRESSION-Sry/Sox
Sry/Sox-IN-TC6 cells

Libbus and Rindflesch ([48]) report on a project that draws on SKR resources to construct a general tool (called GBD) intended to help researchers manage the literature in molecular biology. GBD is designed to processes MEDLINE citations returned by searches to PubMed. A pilot project seeks to identify and extract information regarding the genetic basis of disease. GBD calls on MetaMap to identify diseases and associated clinical findings in the citations retrieved, while the methods of Tanabe and Wilbur ([47]) are used to tag genomic phenomena such as genes, alleles, mutations, polymorphism, and chromosomes. Once such information has been identified in the group of citations returned by PubMed, further processing by GBD determines distributional and cooccurrence patterns for user-selected categories. For example, the user can request a list of genes that cooccur with a specified disease in the output from PubMed; these lists contain links to the citations retrieved, as illustrated in (17).

(17) 20015025|Diabetes Mellitus, Non-Insulin-Dependent|ipf-1
20015026|Diabetes Mellitus, Non-Insulin-Dependent|basal insulin promoter
20015026|Diabetes Mellitus, Non-Insulin-Dependent|insccg243
20053748|Diabetes Mellitus, Non-Insulin-Dependent|g20r
20053748|Diabetes Mellitus, Non-Insulin-Dependent|g385v

Such lists can be generated for cooccurrence of more than one gene with a disease or a combination of genes and clinical findings. The clustering of PubMed output into categories dynamically specified by the user contributes to effective management of the current research literature.

5. Summary

The Semantic Knowledge Representation project seeks to provide usable semantic representation of biomedical text by building on resources currently available at the Library, especially the UMLS knowledge sources and the natural language processing tools provided by the SPECIALIST system. Two existing programs, MetaMap and SemRep, are being evaluated, enhanced, and applied to a variety of problems in the management of biomedical information. These include automatic indexing of MEDLINE citations, concept-based query expansion, accurate identification of anatomical terminology and relationships in clinical records, and the mining of biomedical text for drug-disease relations and molecular biology information.

Current research is investigating the application of SKR resources to applications such as question answering systems, image retrieval, and structured browsing and navigation facilities. The concepts and relationships that underlie the semantic structures produced by MetaMap and SemRep are drawn largely from the domain knowledge contained in the UMLS knowledge sources. Although the UMLS has broad coverage of the biomedical domain, there are gaps, particularly in the area of molecular biology. For example, only about half of a set of disease-related gene names and gene products occurring in the National Center for Biomedical Information (NCBI) database, LocusLink, are found in the Metathe-

saurus. As a way of filling this terminological gap, an effort is underway to augment the domain knowledge available to MetaMap with protein and gene names from LocusLink as well as SWISS-PROT and its supplement, TrEMBL. It is expected that the increased coverage of gene terminology will result in a corresponding increase in the accuracy of the knowledge extraction systems using MetaMap as a basic component ([28][44][45][46][48]).

References

[1] Wilks, Yorick A. 1976. Parsing English II. In Eugene Charniak and Yorick Wilks (eds.) *Computational semantics: An introduction to artificial intelligence and natural language comprehension.* Amsterdam: North-Holland Publishing Co.

[2] Schank, Roger C. 1975. *Conceptual information processing.* Amsterdam: North-Holland Publishing Co.

[3] Riesbeck, Christopher K. 1981. Perspectives on parsing issues. *Proceedings of the Nineteenth Annual Meeting of the Association for Computational Linguistics*, 105-6.

[4] Hahn, Udo. 1989. Making understanders out of parsers: Semantically driven parsing as a key concept for realistic text understanding applications. *International Journal of Intelligent Systems* 4(3):345-93.

[5] Maida, Anthony S., and Stuart C. Shapiro. 1982. Intensional concepts in propositional semantic networks. *Cognitive Science* 6:291-330. [reprinted in Brachman and Levesque 1985]

[6] Brachman, Ronald J., and Hector J. Levesque (eds.) 1985. Readings in knowledge representation. Los Altos, CA: Morgan Kaufman Publishers, Inc.

[7] Bates, M., and Weischedel R. M. 1993. *Challenges in natural language processing.* Cambridge: Cambridge University Press.

[8] Saint-Dizier, Patrick, and Evelyne Viegas (eds). 1995. *Computational lexical semantics.* Cambridge: Cambridge University Press.

[9] Sowa, John F. 2000. *Knowledge representation: Logical, philosophical, and computational foundations.* Pacific Grove, CA: Brooks/Cole.

[10] Haug, Peter J.; Spence Koehler; Lee Min Lau; Ping Wang; Roberto Rocha; and Stanley M. Huff. 1995. Experience with a mixed semantic/syntactic parser. In Reed M. Gardner (ed.) *Proceedings of the 19th Annual Symposium on Computer Applications in Medical Care*, 284-8.

[11] Hripcsak G, Friedman C, Alderson PO, DuMouchel W, Johnson SB, Clayton PD. 1995. Unlocking clinical data from narrative reports: a study of natural language processing. *Annals of Internal Medicine* 122(9):681-8.

[12] Friedman, Carol; Stephen B. Johnson; Bruce Forman; and Justin Starren. 1995. Architectural requirements for a multipurpose natural language processor in the clinical environment. In Reed M. Gardner (ed.) *Proceedings of the 19th Annual Symposium on Computer Applications in Medical Care*, 347-51.

[13] Rassinoux, Anne-Marie; Judith C. Wagner; Christian Lovis; Robert H. Baud; Alan Rector; and Jean-Raoul Scherrer. 1995. Analysis of medical texts based on a sound medical model. In Reed M. Gardner (ed.) *Proceedings of the 19th Annual Symposium on Computer Applications in Medical Care*, 27-31.

[14] Zweigenbaum, P.; B. Bachimont; J. Bouaud; J. Charlet; and J. F. Boisvieuz. 1995. A multi-lingual architecture for building a normalized conceptual representation from medical language. In Reed M. Gardner (ed.) *Proceedings of the Nineteenth Annual Symposium on Computer Applications in Medical Care*, 357-61.

[15] Hahn, Udo; Martin Romacker; and Stefan Schulz. 2000. MEDSYNDIKATE--design considerations for an ontology-based medical text understanding system. J. Marc Overhage (ed.) *Proceedings of the AMIA Annual Symposium*, 330-4.

[16] Lindberg, Donald A. B.; Betsy L. Humphreys; and Alexa T. McCray. 1993. The Unified Medical Language System. *Methods of Information in Medicine* 32:281-91.

[17] Humphreys, Betsy L.; Donald A. B. Lindberg; Harold M. Schoolman; and G. Octo Barnett. 1998. The Unified Medical language System: An informatics research collaboration. *Journal of American Medical Informatics Association* 5(1):1-13.

[18] McCray, Alexa T.; Alan R. Aronson; Allen C. Browne; Thomas C. Rindflesch; Amir Razi; and Suresh Srinivasan. 1993. UMLS knowledge for biomedical language processing. *Bulletin of the Medical Library Association* 1:184-94.

[19] McCray, Alexa T. 2000. Improving access to healthcare information: the Lister Hill National Center for Biomedical Communications. *MD Computing* 17(2):29-34.

[20] McCray Alexa T.; Suresh Srinivasan; and Allen C. Browne. 1994 Lexical methods for managing variation in biomedical terminologies. In Ozbolt JG (ed.) *Proceedings of the 18th Annual Symposium on Computer Applications in Medical Care*, 235-9.

[21] McCray Alexa T.; Amir M. Razi, Anantha K. Bangalore; Allen C. Browne; and P. Zoe Stavri. 1996. The UMLS Knowledge Source Server: A Versatile Internet-Based Research Tool. In Cimino JJ (ed.) *Proceedings of the AMIA Annual Fall Symposium*, 164-8.

[22] Aronson, Alan R.; Thomas C. Rindflesch; and Allen C. Browne. 1994. Exploiting a large thesaurus for information retrieval. In *Proceedings of RIAO*, 197-216.

[23] Rindflesch, Thomas C., and Alan R. Aronson. 1994. Ambiguity resolution while mapping free text to the UMLS Metathesaurus. Judy G. Ozbolt (ed.) *Proceedings of the 18th Annual Symposium on Computer Applications in Medical Care*, 240-4.

[24] Aronson, Alan R. 1996. The effect of textual variation on concept-based information retrieval. In James J. Cimino (ed.) *Proceedings of the AMIA Annual Fall Symposium*, 373-7.

[25] Aronson, Alan R. 2001. Effective mapping of biomedical text to the UMLS Metathesaurus: The MetaMap program. In Suzanne Bakken (ed.) *Proceedings of the AMIA Annual Symposium*, 17-21.

[26] Rindflesch, Thomas C., and Alan R. Aronson. 1993. Semantic processing in information retrieval. Charles Safran (ed.) *Proceedings of the 17th Annual Symposium on Computer Applications in Medical Care*, 611-15.

[27] Rindflesch, Thomas C. 1995. Integrating natural language processing and biomedical domain knowledge for increased information retrieval effectiveness. *Proceedings of the 5th Annual Dual-use Technologies and Applications Conference*, 260-5.

[28] Rindflesch, Thomas C.; Jayant Rajan; and Lawrence Hunter. 2000. Extracting molecular binding relationships from biomedical text. *Proceedings of the 6th Applied Natural Language Processing Conference*, 188-95. Association for Computational Linguistics.

[29] Cutting D.; J. Kupiec; J. Pedersen; and P. Sibun. 1992. A practical part-of-speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing*.

[30] Srinivasan, Padmini. 1996a. Query expansion and MEDLINE. *Information Processing and Management* 32(4):431-43.

[31] Srinivasan, Padmini. 1996b. Optimal document-indexing vocabulary for MEDLINE. *Information Processing and Management* 32(5):503-14.

[32] Srinivasan, Padmini. 1996c. Retrieval feedback for query design in MEDLINE. A Comparison with Expert Network and LLSF Approaches. In James J. Cimino (ed.) *Proceedings of the AMIA Annual Fall Symposium*, 353-6.

[33] Srinivasan, Padmini. 1996d. Retrieval feedback in MEDLINE. *Journal of the American Medical Informatics Association* 3(2):157-67.

[34] Aronson, Alan R., and Thomas C. Rindflesch. 1997. Query expansion using the UMLS Metathesaurus. In Daniel R. Masys (ed.) *Proceedings of the AMIA Annual Fall Symposium*, 485-9.

[35] Wright, Lawrence W.; Holly K. Grossetta Nardini; Alan R. Aronson; and Thomas C. Rindflesch. 1998. Hierarchical concept indexing of full-text documents in the UMLS Information Sources Map. *Journal of the American Society for Information Science* 50(6):514-23.

[36] Aronson, Alan R.; Olivier Bodenreider; H. Florence Chang; Susanne M. Humphrey; James G. Mork; Stuart J. Nelson; Thomas C. Rindflesch; and W. John Wilbur. 2000. The NLM indexing initiative. In J. Marc Overhage (ed.) *Proceedings of the AMIA Annual Symposium*, 17-21.

[37] Weeber, Marc; Henny Klein; Alan R. Aronson; James G. Mork; et al. 2000. Text-based discovery in medicine: The architecture of the DAD-system. In J. Marc Overhage (ed.) *Proceedings of the AMIA Annual Symposium*, 903-7.

[38] Swanson, Donald R. 1986. Fish oil, Raynaud s syndrome, and undiscovered public knowledge. *Perspectives in Biology and Medicine* 30(1), 7 18.

[39] Swanson, Donald R., and Smalheiser, Neil R. 1997. An interactive system for finding complementary literatures: A stimulus to scientific discovery. *Artificial Intelligence* 91, 183-203.

[40] Srinivasan, Padmini. 2001. A text mining tool for MEDLINE. In Suzanne Bakken (ed.) *Proceedings of the AMIA Annual Symposium*, 642-6.

[41] Srinivasan, Padmini, and Thomas C. Rindflesch. 2002. Exploring text mining from MEDLINE. *Proceedings of the AMIA Annual Symposium*, in press.

[42] Sneiderman Charles A.; Thomas C. Rindflesch; and Alan R. Aronson. 1996. Finding the findings: Identification of findings in medical literature using restricted natural language processing. In Cimino JJ (ed.) *Proceedings of the AMIA Annual Fall Symposium*, 239-43.

[43] Rindflesch, Thomas C.; Carol A. Bean; and Charles A. Sneiderman. 2000. Argument identification for arterial branching predications asserted in cardiac catheterization reports. *Proceedings of the AMIA Symposium*, 704-8.

[44] Rindflesch, Thomas C.; Lawrence Hunter; and Alan R. Aronson. 1999. Mining molecular binding terms from biomedical text. *Proceedings of the AMIA Annual Symposium*, 127-31.

[45] Sarkar, I. Neil, and Thomas C. Rindflesch. 2002. Discovering protein similarity using natural language processing. *Proceedings of the AMIA Annual Symposium*, in press.

[46] Rindflesch, Thomas C.; Lorraine Tanabe; John W. Weinstein; and Lawrence Hunter. 2000. EDGAR: Extraction of drugs, genes and relations from the biomedical literature. *Pacific Symposium on Biocomputing* 5:514-25.

[47] Tanabe, Lorraine, and W. John Wilbur. 2002. Tagging gene and protein names in biomedical text. *Bioinformatics*, in press.

[48] Libbus, Bisharah, and Thomas C. Rindflesch. 2002. NLP-based information extraction for managing the molecular biology literature. *Proceedings of the AMIA Annual Symposium*, in press.

Real World Semantic Web Applications
V. Kashyap and L. Shklar (Eds.)
IOS Press, 2002

NESSTAR: A Semantic Web Application for Statistical Data and Metadata

Pasqualino 'Titto' ASSINI (titto@nesstar.com)
Nesstar Ltd - U.K.

Abstract. NESSTAR is a Semantic Web application for statistical data and metadata that aims to streamline the process of finding, accessing and analysing statistical information. The paper describes the rationale behind the design of the NESSTAR system and, more in general, the steps involved in the design and development of a typical Semantic Web application.

1 Statistical Data Dissemination: The Reality and the Dream

The social sciences are big producers and consumers of statistical data. Surveys, censuses and opinion polls are the basic sources for most quantitative research in sociology and economy. These data are collected and preserved by specialised data archives, national statistical offices or private research institutes (e.g. Gallup) and traditionally disseminated to researchers (social researchers, journalists, marketing experts, etc.) as datafiles stored on some form of magnetic media and accompanied by bulky printed documentation (metadata).

From the point of view of a researcher the current statistical data dissemination process is far from optimal. Given that there are many data publishers, each one with its own distinct access and dissemination procedures, it is often not easy to find and get hold of the right information. Also the artificial separation between statistical data and metadata complicates the assessment and processing of the information.

In 1998 the European Union funded a research and development project named Networked Social Science Tools and Resources (NESSTAR). The aim of the project was to bring the advantages of the Web to the world of statistical data dissemination. At the time the WWW had already made the publishing of textual and graphical information easier and cheaper than ever. Huge amount of information had been made available worldwide at a press of a button, at virtually no cost and in a highly integrated way. The question that NESSTAR was called to answer was if it was possible to create a "Data Web" that would make just as easy to publish, locate and access statistical data.

The NESSTAR project has been followed by FASTER [2], another European project that has further developed the Data Web concept and implementation. NESSTAR and FASTER have been sufficiently successful to convince two of their main contractors, the University of Essex in England and the University of Bergen in Norway, to spin off a company to exploit commercially the Data Web technology. Nesstar Ltd is currently busy developing Data Web solutions for a number of international clients.

2 The Semantic Web and the Data Web

The basic aim of NESSTAR is to make available, in an easily accessible way, a great quantity of statistical data and metadata that is currently locked in incompatible or human understandable only formats. If this objective could be achieved it would revolutionize the way people access statistical information just like the World Wide Web has already revolutionised access to other kinds of information.

The WWW has recently known a major evolution with the appearance of a set of new technologies that go under the name of Semantic Web (SW):

> *The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation. [12].*

The necessity of representing statistical metadata and data in a way that is machine-understandable make NESSTAR a perfect candidate for the application of Semantic Web technologies. The NESSTAR "Data Web" is therefore the application of Semantic Web techniques and principles to the problem of distributed data dissemination and processing.

The *modus operandi* of NESSTAR is very simple. Data publishers make their statistical information available as objects, as specified by the NESSTAR domain model, on the Net. The system is fully distributed: each publisher runs its own server.

Users use the system pretty much as they use the Web: if they know where some information is stored they can "point" their client application to it (for example by typing the object URL in a location bar or by clicking on a hyperlink). The client will access the remote statistical object and display it to the user. The users can also perform searches to find object with particular characteristics such as: "find all variables about political orientation". This is similar to using a search engine such as Google to find all HTML pages that contain a given keyword.

The NESSTAR system is built on top of a lightweight Web and object-oriented middleware, the NEsstar Object Oriented Middleware (NEOOM). NEOOM is closely based on Web and Semantic Web standards, in particular RDF [6], RDF Schema [11], HTML and HTTP. So closely, actually, that more than a distinct framework it can be considered as just a set of guidelines on how to use off the shelf (semantic) web technology to build distributed object-oriented systems.

In the following sections I will discuss the major steps in the design and development of a typical Semantic Web Application (SWA) using NESSTAR as a concrete full-scale example. More detailed technical information on NEOOM is available in [8] and in [7].

3 Choosing a Modelling Language

At the core of any SWA there is a formal model of the application domain. The application domain model must necessarily be expressed in a modelling language. The choice of this modelling language is the first significant step in the SWA design process.

The standard SW modelling language is RDF Schema [11] (in the future this might be superseded by a more sophisticated schema language: the Web Ontology Language (WOL) [4]). RDF Schema is an useful tool but as a modelling language has some significant shortcomings:

- limited expressive power (e.g., no relationships, no operations)
- lack of modelling tools (such as graphical editors for RDF Schema models)

For many SWAs a better choice would be the Unified Modelling Language (UML). UML is a standard and expressive modelling language based on the object-oriented paradigm. It is well documented, widely taught and good design tools (both open source and commercial) are available. UML is very extensive and complex but this complexity should not be a deterrent for its adoption as in most cases only a modest subset of the language will be needed to model the application model of a SWA. For example the NESSTAR domain model has been defined using just a simple UML Class Diagram.

4 Modelling the Application Domain

In the case of NESSTAR the application domain model is an object-oriented model of statistical data and metadata[1]. Though not very extensive, it currently contains about 15 classes, it captures many of the key domain concepts: statistical studies, datafiles, statistical variables, indicators, tables, etc. The classes are linked together by a set of relationships: a Study for example contains Cubes each having one or more Dimensions, etc. Some objects have also methods. Statistical Studies for example have methods such as Tabulate or Frequency and other common statistical operations.

In addition to the domain specific concepts the model includes another 10 or so domain independent "support" classes. An example is the Server class. It represents the server where the objects are hosted and provides basic administrative functions such as file transfer, server reboot and shutdown, etc. It also plays, in the Data Web, a role similar to that of the home page in the normal Web. From a home page one can, by following hyperlinks, reach all the contents of a web site. Similarly, starting from a server object an application can, by recursively traversing the object relationships, reach all the other objects hosted by the server. Another generic class is Catalog. Catalogs are used to group objects, just like folders in a filesystem. Catalogs can be browsed, an application can get the complete list of all the objects contained in a catalog, or searched to select only the objects that satisfy a particular condition.

Many statistical studies contain sensitive information that cannot be made available without restrictions. For this reason the model includes a set of objects to represent Users, the Roles they play in the system (example: Administrator, EndUser, DataPublisher), the agreements that they have accepted (such as: "I agree to use these data only for non commercial research purposes"), etc. On top of this small security model is possible to define a variety of access control policies[2].

The domain independent classes of the model are particularly interesting as they can be reused by different SWAs. Hopefully in the near future we will see the emergence of reusable 'libraries' ("ontologies" in SW-speak) of domain independent concepts. That will greatly speed up the process of modelling and implementing new SWAs.

[1] For reasons of space the object model class diagram is not included in this paper. It can be found at http://www.nesstar.org/sdk/nesstar_object_model.pdf

[2] NESSTAR servers support a wide range of Access Control mechanisms.

5 Publishing the Model

Once the model is defined it is useful to make it available on the Net. By doing so it becomes possible to build highly generic applications that discover object characteristics (properties, relationships, methods, etc. as specified by the object class definition) dynamically at run-time. For NESSTAR we have developed one such tool: the Object Browser, a web-based generic client used for object testing and administration. The Object Browser can be used to display and operate on any NESSTAR object.

To publish the model on the SW it has to be converted to an equivalent model expressed in the SW standard modelling language: RDF Schema. Mapping a UML Class Diagram, or other simple object-oriented formalisms, to RDF Schema is relatively straightforward. RDF Schema provides all the basic object-oriented primitives: classes, properties and inheritance. There is only one major omission: RDF Schema does not provide a way of describing the behaviour of an object, that is to say the operations that it can perform.

5.1 Specifying Behaviour

The absence in the SW of a standard way of representing behaviour is a major problem. Another technology, known as Web Services, has recently stepped in to fill the gap. The downside of this otherwise very positive development is that, as Web Services and the Semantic Web have been defined by different organisations, they are not well integrated.

Luckily the standardization process of the Web Services Description Language (WSDL) [10] (the part of the Web Services proposal that deals with the specification of services) has now moved to the W3C. As part of this activity the W3C should soon produce a standard model to describe object behaviour in RDF Schema.

For NESSTAR we could not wait for this standardization process to complete. The NESSTAR objects can perform a wide range of operations: queries, statistical operations, file transfers, etc. and we needed a way of specifying them formally. In order to do so we have defined the NEOOM Object Model, a small 'RDF ontology' to describe methods (see [7] for details). With this extension RDF Schema becomes a fully-fledged Interface Definition Language (IDL).

In the NEOOM Object Model (see Fig.1), methods (e.g. *Login*) are defined as subclasses of the *Method* class. Method invocations are represented by instances of the method classes. Defining a method as a class is a bit unusual [3] but it has the advantage of making a method invocation an object in its own right. Being an object a method invocation can be represented in RDF (and therefore stored, transmitted or logged easily) and can have methods and properties.

Method parameters are defined by instances of the *Parameter* class. A *Parameter* is conceptually very similar to an *rdf:property* and it inherits from it. As methods are classes then a parameter is just one of its instance properties.

[3] In Java for example, a method is represented by an *instance* of *nesstar.lang.reflect.Method*, not by a separate class.

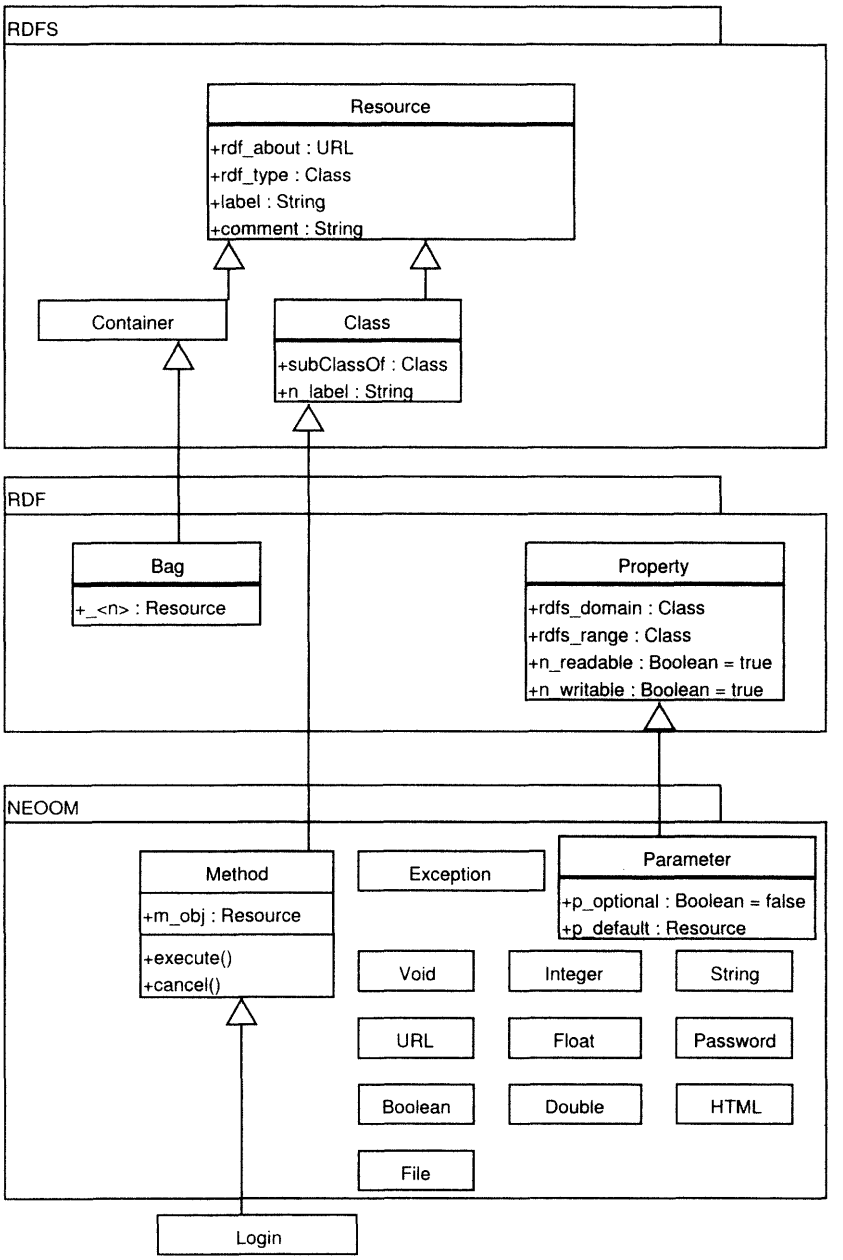


Figure 1: NEOOM Object Model

6 Managing Model Changes

No matter how good a SWA's domain model it will never be complete and final. Information publishers will always need to extend it to represent more specific information or to add new functionality. These extensions will often be applied by different organizations and in an uncoordinated way.

The long-term viability of a SWA depends on its ability of accommodating these extensions gracefully so that they do not lead to a "balkanisation" of the system.

RDF and RDF Schema have been designed with distributed extensibility in mind and provide a variety of mechanisms at this effect: class and property inheritance, open-ended set of class properties, reification.

Using inheritance is possible to extend and specialize an existing concept without breaking all the applications that depends on it. We have had a good demonstration of the power of this mechanism recently when after having massively extended the NESSTAR object model our old clients have (mostly) kept on working by treating the new more specialised objects of the new model as the more general concepts they were originally developed to process.

Incidentally this is one major advantage of RDF Schema with respect to WSDL [10] as an IDL. WSDL not being object-oriented, basically is just an RPC specification language, does not support the same kind of interface extensibility. This will make it difficult to upgrade and evolve Web Services applications (an issue that might soon disappear if WSDL will be redefined as an RDF model).

The fact that the modelling language is extensible unfortunately does not automatically guarantee that any software application built on top of it will be able to gracefully handle future extensions. Applications developers can easily commit the mistake of relying on aspects of the model that will change and be invalidated at a later time.

7 Choosing a Representation Format

Once the model has been defined it is possible to describe and publish the actual instances of the objects (e.g. a Dataset, a Variable) on the SW.

To do so we need to choose a suitable representation format, a syntax that we will use to code our objects in a machine-parsable form.

The SW provides a standard solution to this problem. The objects are described in RDF [13] and coded in the standard RDF's XML representation.

A possible alternative would be to define a specialised, *ad hoc* XML syntax.

A possible advantage of this solution is that, if properly done, it could lead to a more compact or simpler syntax.

The disadvantages are far more significant though: a good syntax is difficult to design properly, a specialised syntax requires a specialised parser and being non-standard will make it harder the integration with other SWAs.

For these reasons the NESSTAR system uses the standard RDF's XML syntax.

8 Publishing Information on the Semantic Web

The XML-coded objects can be distributed in a number of different ways: stored in a file, sent by email, published on a Web site, etc.

For a SWA is particularly important to specify how the objects are going to be made available on the Web.

There is a very simple solution to this problem. Being so simple it deserves a pompous name, we might call it the Self-Description principle:

> *Semantic Web Objects and Classes self describe themselves by making accessible, via HTTP or other protocols, their RDF description at their URL.*

This is nothing new. The principle simple states that SW objects are accessed exactly as any other WWW resource. This is also the solution that we have adopted in NESSTAR. Each NESSTAR object "lives" at some HTTP URL. When a user client accesses the object URL the object returns a description of its current state in RDF.

8.1 Performing Remote Method Calls

Finally we need to specify how to perform remote object-oriented calls so that remote clients can access objects behaviour.

SOAP [9] is an increasingly popular solution to this problem. In NESSTAR, even if we plan to support SOAP in the future, we currently rely on an older and much simpler alternative: the protocol used to submit HTML Forms (as specified in [14, sect17]). It is very easy to define a set of conventions to map method calls on top of this simple protocol (again see [7] for details). This solution has a couple of significant advantages with respect to sending a SOAP XML message: method calls can be performed using a normal web browser (and in general using programs written in any language that comes with an HTTP library) and it is easy to determine an URL that corresponds to the operation. The URL can be used by an application to represent and replay the operation.

9 NESSTAR System Components

The basic architecture of the NESSTAR system is identical to the WWW architecture. Resources are hosted on NESSTAR Servers. The Servers serve both normal WWW resources such as HTML pages, images, etc. and statistical objects. Just as the WWW, NESSTAR is at the same time fully distributed, each server is totally independent and there is no single point of failure, and integrated, as users experience it as an interconnected whole.

Users can access the system using the NESSTAR Explorer (a Java application for power users), the NESSTAR Light Explorer (a WWW interface based on Servlet technology), the NESSTAR Publisher (a metadata editing, validating and publishing tool) or the Object Browser (a web based generic client used for object testing and administration).[4]

The NESSTAR Explorer is particularly interesting for its similarity to a common Web Browser. Users can enter the URL of any WWW or NESSTAR resource in a Location bar. Normal WWW resources will be displayed just as they would in a normal Web Browser (or an external Web Browser is invoked to handle them), NESSTAR statistical objects are handled specially and displayed and manipulated in an efficient and flexible user interface.

[4]The clients can be downloaded from the NESSTAR Web Site [3]. Sample Web clients (the Object Browser and the NESSTAR Light) can be accessed at http://nesstar.data-archive.ac.uk/

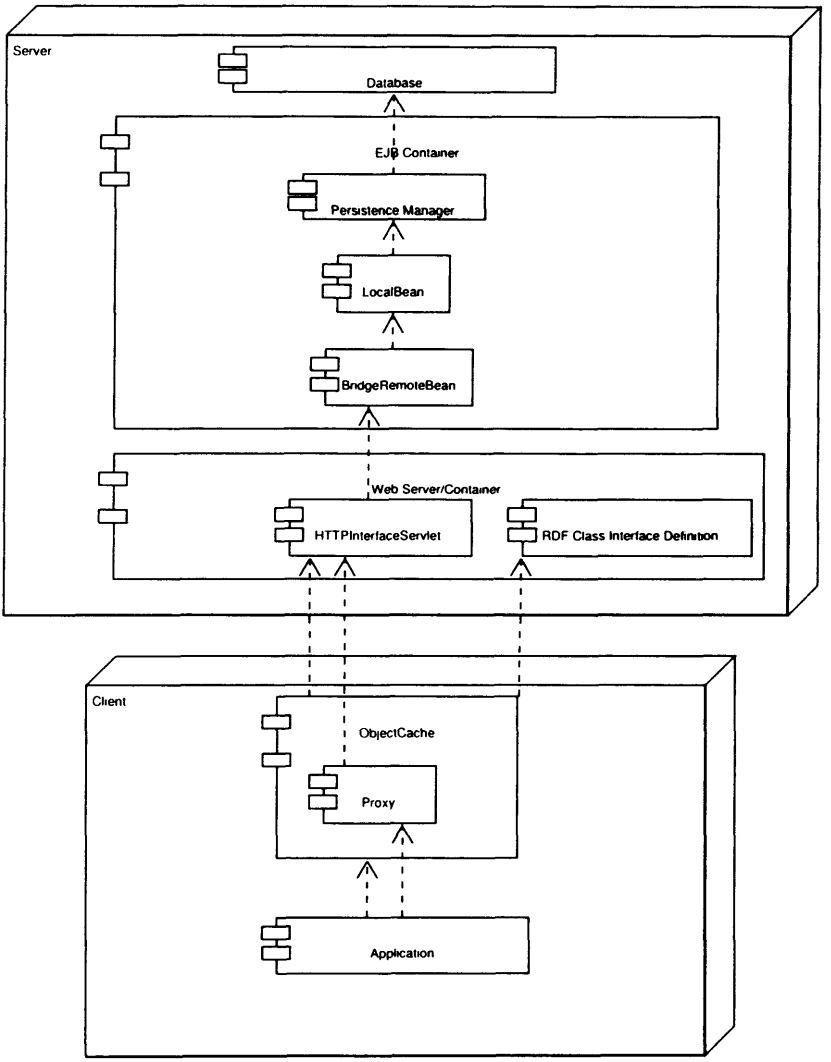


Figure 2: System Components

The current version of NESSTAR is mainly implemented in Java (some modules are in C++). The deployment diagram 2 shows the main components of the system. On the server side we have three main components: an Enterprise Java Bean Container [1] (JBoss 2.4.x in the current implementation), a Web Server/Container (currently Tomcat Catalina) and a relational database. The NESSTAR objects are hosted in the EJB Container as Beans. They have only local interfaces so they are not directly accessible from the outside of the container. Access takes place through a Bridge Bean. The interface with the Web is implemented by a Servlet that converts HTTP requests in RMI calls to the Bridge Bean. The RDF class interface definitions are stored in the Web Container. They are downloaded on demand by clients, such as the Object Browser, that need to discover at run-time the interface of an object.

Java client side applications access the remote NESSTAR objects through an object oriented API that supports:

- execution of remote object methods
- access of remote object properties
- traversing of object relationships
- operation bookmarks
- caching of remote objects
- handling of (multiple) authentication challenges
- HTTP and HTTPS wire protocols

For every class of NESSTAR objects the API includes a corresponding 'proxy' class. All accesses to remote objects take place through the proxy classes. The proxies are hold in an object cache. When a client application asks for an object with a given URL the API checks if the object is already in the cache. If this is not the case it performs an HTTP GET operation to the object URL. If an RDF description is found at the URL the corresponding proxy instance is automatically created, cached and returned to the calling application.

Once an application has got the proxy corresponding to the desired object it can access its properties via normal accessor methods (get/set) and perform operations on it. For each operation the API provides a corresponding URL. The application can store the operation URL so that the operation can be replayed at a later time.

10 Design and Development in NESSTAR

The design and development of NESSTAR objects is rather straightforward (see the use case diagram Fig.3). A Designer uses an UML modelling tool (currently we use Argo/UML or Poseidon) to create a Class Diagram specifying the classes to be created. The class diagram is saved as an XMI [5] file, the standard format for storage of UML diagrams. The XMI file is processed, using XSLT scripts, to generate for each class:

- the Interface definition in RDF
- the Java proxy for the API

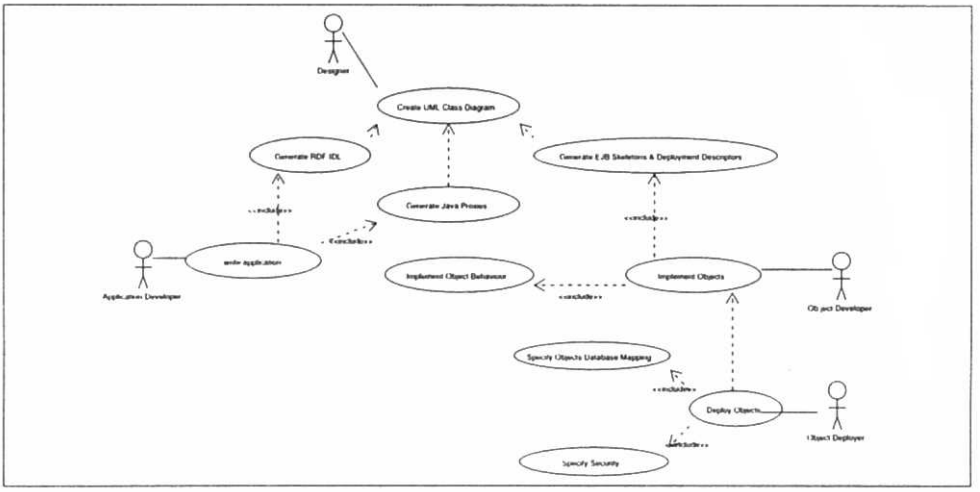


Figure 3: Design and Development Use Cases

- the Skeletons of the Enterprise Java Bean plus the corresponding EJB deployment descriptor

The Object Developer can now proceed to implement the object by adding the methods implementations to the object skeletons. A Deployer completes the process and deploys the new classes in the EJB server by specifying their database mapping and security information.

The Application Developer adds the proxy classes for the objects he is interested in processing to its classpath and use them to write his application. Applications can ask the API to read in the RDF Interface Definitions to discover dynamically the properties and operations of any NESSTAR object.

11 Conclusions

The advent of the Semantic Web has made possible the creation of a new class of distributed applications that combine the simplicity and scalability of the Web with the formal interfaces and modelling of traditional object-oriented middleware.

SW technology offers many benefits:

- Support for sophisticated domain models
- Distributed extensibility
- Integration with the current Web
- Programming language independence

NESSTAR uses the strengths of both the current WWW infrastructure and the new Semantic Web technologies to provide at least some of the basic element of the Data Web "dream".

Compared with other Semantic Web applications NESSTAR is rather simple. It doesn't define sophisticated ontologies or make use of advanced SW features such as reification or logical inference. The fact that even simple SWAs as NESSTAR can deliver real value to information publishers and users is certainly a good omen for the future of the Semantic Web vision.

References

[1] Enterprise JavaBeans. http://java.sun.com/products/ejb/.

[2] EU Project FASTER. http://www.faster-data.org.

[3] EU Project NESSTAR. http://www.nesstar.org.

[4] Web Ontology Language (WOL) Working Group. http://www.w3.org/2001/sw/WebOnt.

[5] XML Metadata Interchange (XMI). http://www.oasis-open.org/cover/xmi.html.

[6] Resource Description Framework (RDF) Model and Syntax Specification. http://web4.w3.org/TR/REC-rdf-syntax/, February 1999.

[7] Pasqualino Assini. NEOOM: A Web and Object Oriented Middleware System. http://www.nesstar.org/sdk/neoom.pdf, 2001.

[8] Pasqualino Assini. Objectifying the Web the 'light' way: an RDF-based framework for the description of Web objects. In *WWW10 Poster Proceedings*, Hong Kong, May 2001. World Wide Web Consortium.

[9] Erik Christensen et al. *Simple Object Access Protocol (SOAP) 1.1.* World Wide Web Consortium, 2000.

[10] Erikothers Christensen, editor. *Web Services Description Language (WSDL) 1.1.* World Wide Web Consortium, March 2001.

[11] R.V. Guha and Dan Brickley, editors. *Resource Description Framework (RDF) Schema Specification 1.0.* World Wide Web Consortium, March 2000.

[12] James Hendler, Tim Berners-Lee, and Ora Lassila. The Semantic Web. *Scientific American*, May 2001.

[13] Ora Lassila and Ralph R. Swick, editors. *Resource Description Framework (RDF) Model and Syntax Specification.* World Wide Web Consortium, 1999.

[14] Dave Raggett et al., editors. *HTML 4.01 Specification.* World Wide Web Consortium, December 1999.

Real World Semantic Web Applications
V. Kashyap and L. Shklar (Eds.)
IOS Press, 2002

Collaborative Decision Making and Personal Knowledge Management with R-Objects Pepper™

Johannes ERNST
R-Objects Inc.
http://www.r-objects.com

Abstract. The process of decision making in distributed groups is described, and requirements for supporting software are derived. An application to support this process, R-Objects Pepper, is discussed from an architectural, and a user's point of view. Pepper is based on a strict semantic information model, and the advantages of this approach are outlined for the requirements posed by distributed decision making. An issue with the emerging "mainstream" semantic web standards for similar applications is raised.

1. Introduction

How do you make decisions? By thinking about the problem for a little while and then deciding in favor of "what feels right?" Does the way you make a decision depend on how important the decision is? Have you ever overlooked critical information at the time when you originally made the decision and why did you? Do you ever have to go back, re-think, and then decide differently, and if so, how did you do that?

How do groups that you belong to make decisions? Not at all? (That's certainly true for some groups) Haphazardly, with lots of coffee, hard negotiation and/or eventually agreeing on compromises that don't meet anyone's goals? Or acquiescing to Joe's proposal, because he yelled loudest? Or very effectively, with no bruised egos, so most members support most group decisions most of the time?

In spite of all the trouble with decision making in practice, most decision making follows a fairly simple base process. While iterations are normal, its key components are, in sequence:

1. **Framing the problem** or question that requires a decision or an answer (example question: "How can we increase revenue in the next fiscal year?")
2. **Collecting** as much **information** as it appears reasonable which can help making a better decision (in this example: sales projections, competitive intelligence, information about new products etc.)
3. **Generating ideas** for alternatives, or proposals for potential courses of action (e.g. opening up an office in a neighbor country, reducing manufacturing cycle time etc.) and analyzing their impact
4. Selecting a **measure of "goodness"** (i.e. quality, sometimes called "cost" in the context of optimization) by which the merits of the ideas can be evaluated: on an absolute scale, and against each other (e.g. impact on the sales forecast, risk, environmental compliance, or a mixture)
5. **Deciding** to pick that one of the previously generated ideas that is "best" according to the selected measure of goodness.

Complex decisions may often involve many more steps, sub-decisions, side conversations and many other items. On the other hand, even in case of simple decisions, if any of the above items are missing in a decision making process, the resulting decision is most likely going to be sub-optimal, and, in a group scenario, often very contentious. For example, if you selected idea A over idea B because it saved your company money, then you will likely face unhappy management if those managers gave you the objective to increase market share, not to lower cost (incorrect, or missing step #5, "selecting a measure of goodness").

Is there a way to provide electronic tools (i.e. software) which can help us making decisions? To analyze this, it may be useful to imagine the following, not-so-uncommon situation: a group needs to make a decision, and for various reasons, it is not possible to assemble everyone in the same meeting room and lock the doors until the decision has been made, which is one time-tested way groups can be forced to actually make decisions. For example, many open source development teams have never met face-to-face.

What kinds of software tool would you ideally like to use in order to facilitate such a collaborative decision making process in order to arrive at the best decision, in the shortest possible amount of time, with the strongest support by as many members of the group as possible, while also enabling new members of the group to easily understand why a certain decision was made one way and not another, so many months later? Here are some of the requirements for such a software system:

- It must be networked as close to real-time as possible, so people don't argue based on out-of-date information
- It must be collaborative, so people can discuss and work on specific aspects of the overall problem, and then integrate their results
- It must support "transient data", so people can "talk about" any issue. Also, it must support more "permanent data", such as the key factors and arguments influencing the decision. The more permanent data needs to be archived so the rationale for the decision can be reconstructed later. The more transient data should either not be archived, or at least be archived in a different place, so that it does not get in the way of using the archive effectively (which is often a problem with archived mailing lists or newsgroups).
- Most importantly, it must aid in the above decision making process, so people understand whether a given comment is a proposal or a consequence, a measure of goodness or supporting data etc. In other words, it needs to support and "understand" the semantics of the decision making process.

At R-Objects Inc., we have developed a software system that, among other things, supports individuals and (potentially distributed) groups in decision making. It is constructed using a hybrid of web services and peer-to-peer network technology as well as a broadly extensible semantic information model. We consider Pepper an (admittedly somewhat unorthodox) application of the semantic web, and will discuss its architecture and use in the context of the semantic web in the remainder of this article.

2. Pepper Architecture

2.1 Overview

When users run Pepper for the first time, they access the product through the Pepper product website to which registered users have access (see Figure 1). Through Java Web Start, Pepper is downloaded and installed on the user's local PC or workstation. Pepper automatically registers on the network, so other users can find this instance of Pepper.

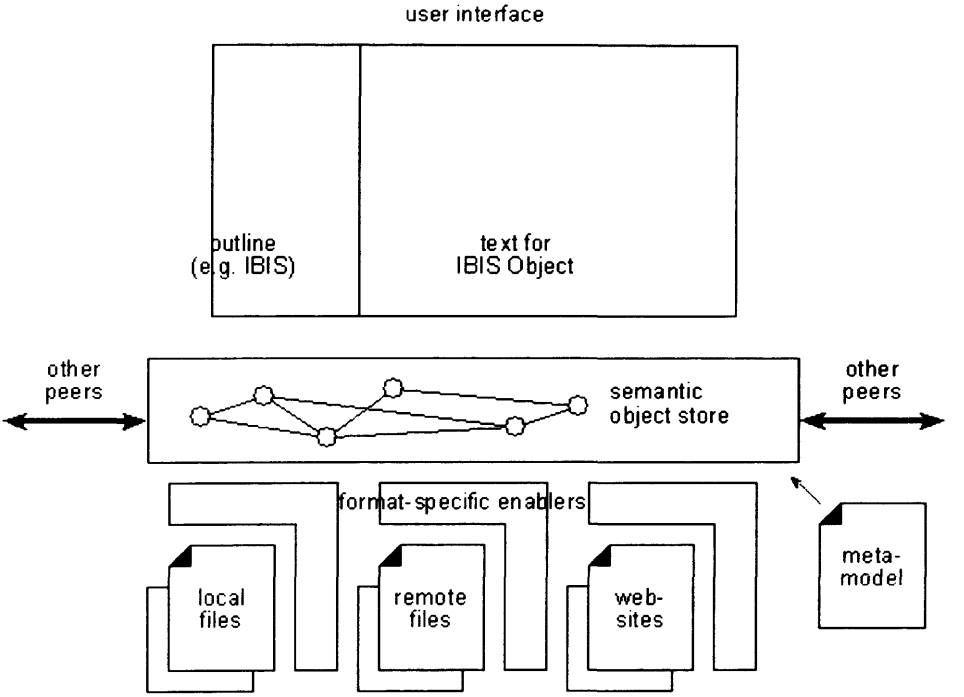


Figure 1: Pepper architecture overview

As Pepper is built using a configurable, semantic, meta-model, it is appropriate to first discuss data in Pepper in general, and only later examine the particular types of data that Pepper supports today.

2.2 Distributed Data Management in Pepper

Data managed by Pepper can come from several sources. For one, a Pepper user can create native Pepper data through Pepper's user interface. Secondly, the user can drag-and-drop data from outside Pepper into Pepper. For example, the user can drag-and-drop a web page from an industry-standard web browser into Pepper, or drag-and-drop a file that is local on his computer from the computer's desktop.

As soon as an external data source has been dropped into Pepper, Pepper parses the data at that data source based on an extensible directory of available parsers (we call them "enablers"), and then presents it to the user through one or more of the available Pepper viewlets (see below). This is shown in the lower segment of figure 1.

Data that has been brought into Pepper from the outside has two important, somewhat unusual characteristics:

- Pepper keeps a link back to the original data source and checks for updates. If a file or web page is being changed outside of Pepper after it has been imported, Pepper will notice that it has changed and forward the appropriate changes, as fine-grained events, to all affected user interface components, and to subscribers of its content on the network. If the old and the new version of a file differ only in one paragraph, for example, a single change event is generated that indicates that single paragraph's change, leaving the other paragraphs (in Pepper) unchanged.

- Pepper does not just shuffle opaque blocks of data (aka files) around (it can, but that's a somewhat less interesting use). Instead, the particular enabler in the enabler directory that is applicable to this particular type of imported data source breaks down the file into individual chunks of information (we call those Peppercorns(tm)) and semantic relationships between those Peppercorns, depending on the semantics of the data source. For example, if an HTML file were imported into Pepper, Pepper would import it as a set of related Peppercorns. (Imagine these Peppercorns and their relationships as close to the file's Document Object Model (DOM)). This will be important below. Through the extensible set of enablers in Pepper, enablers appropriate to different types of data sources can be invoked, preserving the semantics of the data source as much as possible. At this point, we have developed enablers for several XML schemas, HTML and several proprietary file formats.

In Pepper, any Peppercorn (i.e. fine-grained data object) has a type, a corresponding set of attributes, and a set of relationships that it can enter based on that type. In fact, all data within Pepper, whether native data or imported from the outside, is represented as strict instances of a broadly extensible semantic information model (called a meta-model). The user can relate any two Peppercorns based on the rules established by the meta-model, but these rules are strictly enforced. Of course, semantically weak 0..*/0..* "catch-all" relationships can be defined for situations where strict enforcement may not be desirable.

The ability of the user to relate any two Peppercorns is NOT limited by whether or not those Peppercorns come from the same or different data sources (e.g. file or URL), or whether they are native in Pepper. This means, for example, if one Peppercorn has been imported from a network data source (say, an XML data stream) and another has been created locally in Pepper, the user can relate them as long as the meta-model permits it. As a result, the layer of Peppercorns and relationships in Pepper is an abstraction layer that represents all Pepper-enabled data in a uniform semantic object network, regardless what format it is, where if came from, or who created them. Because Pepper checks back regularly with the imported data source and forwards fine-grained, event-based updates, relationships across data source boundaries remain in-place even if the imported data is being changed from outside of Pepper.

Unlike basic XML, Pepper's meta-model is not limited to tree structures but is a specifically developed variation of a powerful, conceptual extended entity-relationship-attribute meta-modeling technique. Its static capabilities support typed (meta-)entities, typed (meta-)relationships, single and multiple inheritance of meta-entities, inheritance of meta-relationships, multiplicity constraints of the form a..b for meta-relationships (with a,b: 0..N), both meta-attribute and meta-role refinement and restriction, default values etc. In addition, dynamic (aka "projected") meta-entities and meta-relationships can be constructed that allow the user to create dynamic, object-oriented, auto-updating views of the underlying data, regardless of whether it is native in Pepper or imported from the network (this latter, although interesting, feature is not further discussed in this article).

Peppercorns and relationships are stored in an object-oriented repository with advanced, object-oriented data replication features based on object leases and distributed locking. Network transport can use one or more of common protocols, such as CORBA, Java/RMI, HTTP and JXTA. This means that user A can make accessible to user B an object that is currently stored in A's object repository. Pepper will automatically replicate that object to the repository of user B. After the initial replication, Pepper will keep the two replicas in sync until B does not use the object any more. It is important to point out that if users A and B want to share one object over the network, this does not imply that they need to replicate all of their objects. Pepper keeps these two overlapping object graphs in sync without placing any additional burdens on the user.

2.3 Viewlets

On top of Pepper's data management capabilities, Pepper provides a component-based user interface that is driven by a directory of available user interface components (we call them Viewlets). The set of viewlets is an open-ended list and there are no compile-or run-time dependencies of the Pepper kernel on any of the viewlets, making viewlets developed by 3rd-party full citizens of the Pepper framework, which has been one of the design goals of the Pepper framework.

A Viewlet comes with a self-announcement as to which types of data they can work on. Consequently, when a user selects a certain Peppercorn in the Pepper GUI, the Pepper GUI self-assembles itself based on the appropriateness of certain user interface components for the selected objects. The underlying Viewlet framework shall not be discussed further in this article: suffice it to say that this meta-data-driven approach to user interface construction allows the construction of completely interoperable, but independent user interface components even by third parties, and incremental additions to other people's user interfaces. This allows 3rd parties (e.g. independent software developers incorporating Pepper technology into their products, or IT departments building company-specific applications) to quickly develop advanced, collaborative, semantic, distributed applications.

3. Semantics in Pepper

It was mentioned previously that all data is typed in Pepper, typically fine-grained and related according to a semantic meta-model. This semantic meta-model currently provides close to 100 meta-entities (aka classes), several dozen meta-relationships (aka associations) and hundreds of meta-attributes. It covers domains from structured decision making, the subject of this article, over requirements management, a simple form of project management, dictionary concepts, to the control models used in embedded real-time software development. It is an Integrated Meta-model, which means that the meta-model supports seamless cross-relationships between different domains: after all, most knowledge work and decision making is cross-disciplinary in one way or another.

To best explain the semantic meta-model in Pepper, we will use the Pepper IBIS meta-model as an example. IBIS is a structured discussion, decision making, and facilitation technique originally developed in the 1970's [7], which is implemented by Pepper in a collaborative fashion. The core concepts of IBIS are the following:

- **Question:** the question, or problem, that needs to be answered and solved. This corresponds to step #1 above.
- **Idea/Proposal for an Answer:** a generated idea that can solve the problem or answer the question (#3 above).
- **Argument in favor:** an argument in favor of a particular Idea (subjective version of #4)
- **Argument against:** an argument against a particular Idea (subjective version of #4)

Further, any IBIS object can be challenged by a new question, and any IBIS object can be backed up by additional information (such as a quote from a document outside of the scope of base IBIS).

The Pepper meta-model for IBIS is shown in Figure 2. The graphical representation of the meta-model here roughly follows the UML graphical syntax. Its mapping to the described IBIS concepts above is straightforward. IBISObject is a common, abstract supertype of all IBIS-related meta-entities in Pepper. For the meaning of SemanticInformationObject and HTMLSection, see below.

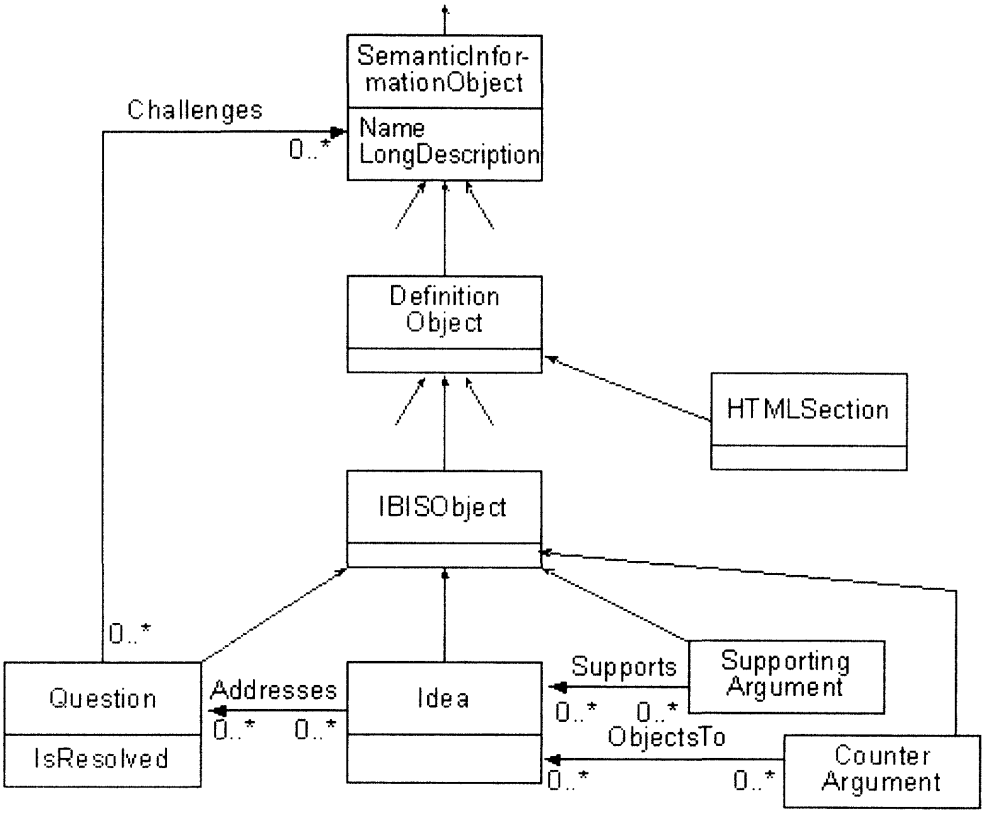


Figure 2: IBIS Meta-model in Pepper

In Pepper, this meta-model is expressed in XML. To make Pepper use this (or any other) meta-model, a code generation process is being employed as part of the nightly build which generates Java interfaces and classes that implement the meta-model in a form that is familiar to the Java programmer. For example, a part of the generated code for the IBIS Question meta-entity above appears in Java as follows:

```
public interface Question extends IBISObject {
    public void setName( StringValue newName )
        throws DoNotHaveLockException, TransactionException;
    public StringValue getName();
    ...
}
public class MemoryQuestion extends MemoryRootEntity implements Question
{
    public void setName( StringValue newName )
        throws DoNotHaveLockException, TransactionException
    {
        ...
    }
    public StringValue getName()
    {
        return theName;
    }
    ...

    public void addPropertyChangeListener(
            PropertyChangeListener newListener)
    {
```

```
        ...
    }
    ...
}
```

The generated code supports applicable Java conventions (property naming, Java Bean-style event generation etc.) and is mostly transactional in nature. We distinguish cleanly between interface and implementation classes so that no viewlet that uses the meta-model ever depends on the details of the code generation process or the data management and replication protocols. In other words, all viewlets work on the interfaces only, never on the implementation classes (shown above is one possible implementation, an in-memory implementation, indicated by the Memory prefix of the generated implementation classes). This frees the Pepper application programmer from having to know about the rather complex mechanisms of our distributed object management, and consequently, enables developers to build novel, collaborative, semantic applications probably more quickly than using any other approach or technology that we are aware of.

Going back to figure 2, notice that IBISObject inherits from a supertype called DefinitionObject, which in turn inherits from a supertype called SemanticInformationObject. SemanticInformationObject is a common supertype of all semantic object types in the meta-model in Pepper across all domains. The meta-relationship between Question and SemanticInformationObject implies that in Pepper, any semantic object can be challenged using an IBIS-style Question, which is ideal for collaborative knowledge work. DefinitionObject is also a supertype of HTMLSection, which is the concept used to represent paragraphs (and similar elements) in HTML documents when they are imported into Pepper. Because of that common inheritance, IBISObjects and HtmlSections share certain behavior, and can be related to other objects in a similar way. Without explaining in detail how this works, it allows, for example, IBISObjects to contains HTMLSections (from other documents) as backup material, addressing step #2 above and also illustrating the interdisciplinary nature of Pepper. Because HTMLSection is a subtype of SemanticInformationObject, and thus inherits the ability to be challenged by an IBIS-style Question, this allows users to perform in-place, semantically assisted IBIS problem solving around contentious paragraphs on any web page, for example. Recall that this ability is independent of whether the HTMLSection has been created as a native Pepper object, or imported (with a link back) from a web page, or replicated over so many copies of Pepper communicating using Pepper's networking protocols.

4. Usage scenario

Using a series of actual Pepper screen shots (starting with figure 3), let's go through a scenario how to use Pepper with the IBIS semantics for collaborative decision making.

In step #1 of the process outlined above, user A phrases the problem to be solved. To do this, A instantiates an object of type Question in Pepper. This Question is shown in figure 3 in the left pane. The right pane contains all viewlets that can edit or show the Question object, and are commonly used for editing and to show details.

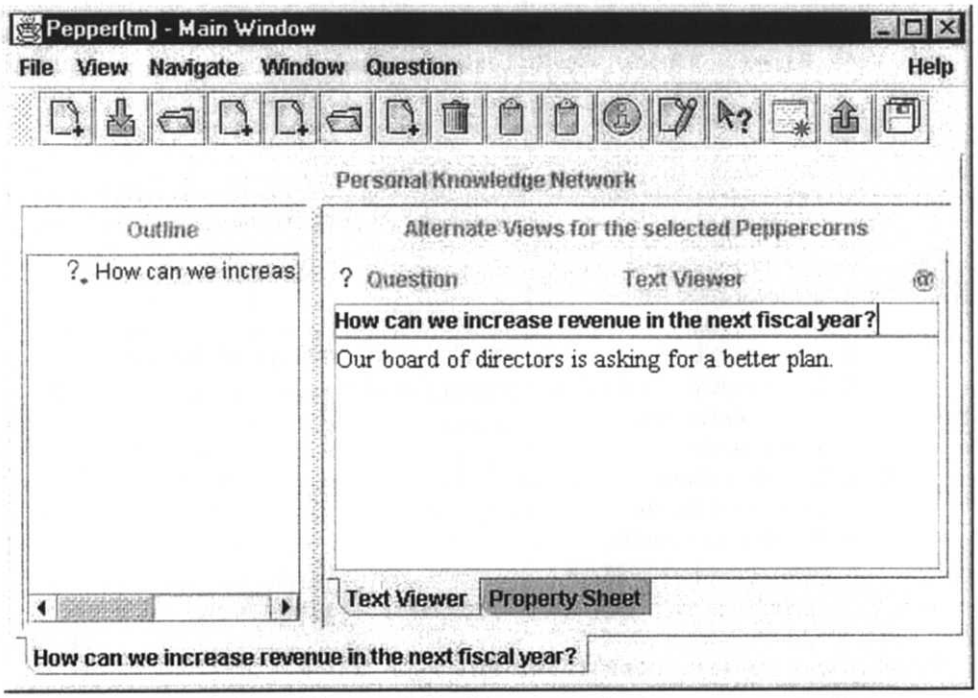


Figure 3: Posing an IBIS Question in Pepper

After a user has posed this question, it can be replicated to the affected work group on the network by a simple click of the mouse, giving everyone the exact same, live, view, on the ongoing IBIS discussion. Individuals can now make suggestions by selecting from a semantics and context-aware dialog. For example, user B may have an idea how to answer the question. In Figure 4, he picks a new "Idea/Proposal for an Answer" that he intends to propose as an answer for the Question.

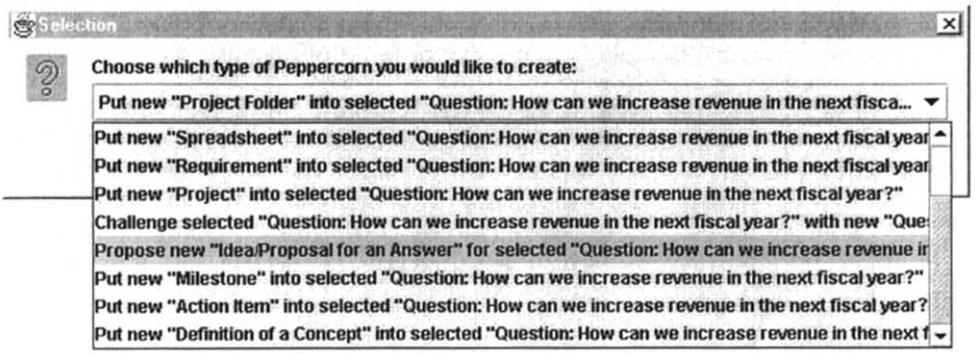


Figure 4: Creating a semantic Peppercorn (new Idea/Proposal) related to a current Peppercorn (selected Question)

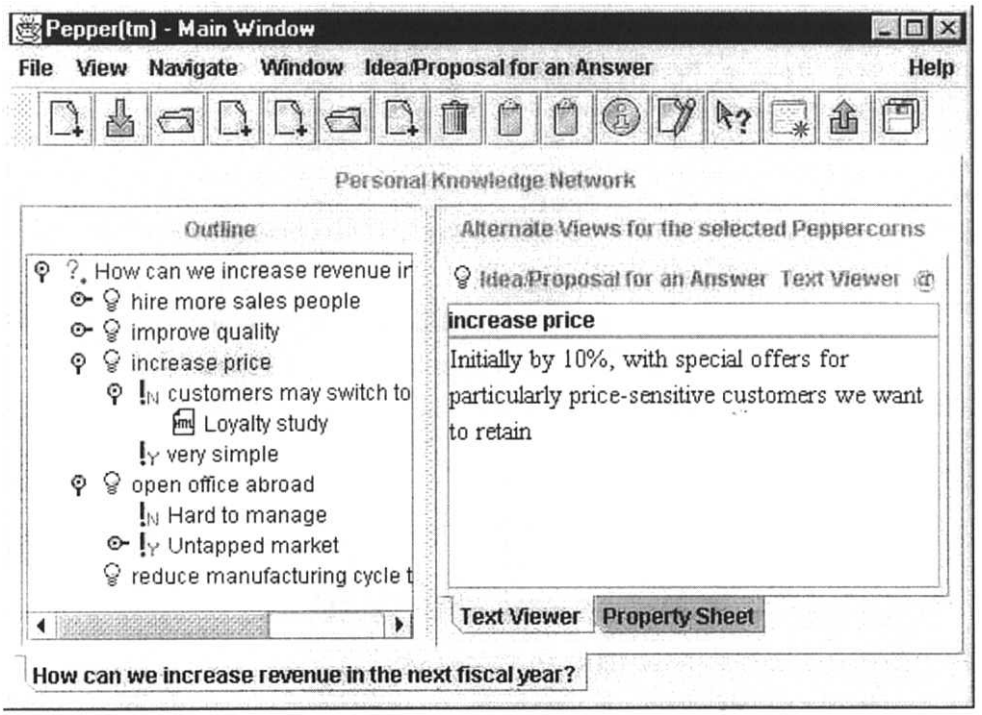


Figure 5: IBIS outline in Pepper with Question Peppercorns, Idea Peppercorns, SupportingArguments and CounterArguments as well as supporting information in HTML

After some time of collaborative editing, discussion and annotation, assembling more Ideas and many arguments in favor and against, plus linking in supporting material from the web or from the users' local hard drives, the IBIS outline may look as shown in figure 5.

For the more transient data in collaborative work, Pepper provides industry -standard compatible instant messaging with the ability to reference arbitrary Pepper Peppercorns from within the instant messaging window, thereby facilitating the IBIS discussion further. E-mail is integrated in a similar way.

The semantic meta-model in Pepper clearly support this decision making process better than any software tool could that was not aware of the semantics inherent in this process (for example, it is very hard to "sort out" heated discussions performed through non-semantic e-mail or threaded discussions, as we probably all know from experience).

Not only can users comb through this discussion much more quickly, and discuss the subject much more orderly and to the point, but given that the semantics are represented explicitly in Pepper and due to the extensible nature of the Pepper framework, one can now develop smart "agent"-type functionality that takes advantage of understanding "what is what". For example, an agent could easily determine how many open issues there are in a current project, how many of those have no proposals, or only proposals with many more arguments against than in favor, if only for management reporting purposes. We have successfully implemented functionality of that type. There are also very interesting opportunities to link in enterprise applications, and/or autocategorization technology.

5. Summary and Further Thoughts

This article focuses on the technology of R-Objects Pepper and its use of semantic concepts for decision making. Using Pepper clearly shows the advantages of semantic, software-supported concepts in personal knowledge work. In this respect, we very much agree with Jon Bosak's infamous "I want my data back" as a case for making the web semantic – after all, how is a computer to support a human if it has no chance to understand the data that it deals with?

Taking a somewhat broader view, this example use of Pepper probably illustrates many of the challenges and potential solutions for developing semantic web technology and applications for similar or even very different purposes. For example, it has been observed often (e.g. [6]) that for the semantic web to meet its promise, it needs to support decentralized authoring of content, but also the decentralized creation and evolution of schemas, which, as it is usually argued, implies that links can only be one-directional, "404's" are allowed and common, and close-to-real-time referential integrity is impossible to achieve on the web.

On the other hand, as the IBIS example shows, links between semantic web objects (aka Peppercorns) must really be presented to the user in a bidirectional manner for many applications: if a Pepper user was forced to decide, for example, whether he should allow others to either allow traversal from the Idea to the SupportingArgument so they could understand the arguments in favor of an idea, or from the SupportingArgument to the Idea so he can understand what Idea a given Argument supports, but not both, any run-of-the-mill business user is most likely to throw away that "application" as it would be unusable for any business purpose. For group decision making it must be possible that different authors create different nodes in the IBIS hierarchy, and updates are communicated in close to real time, however.

Consequently, we clearly believe that the assumption of one-directionality of hyperlinks in the semantic web cannot be held up in many, similar, collaborative real-world usage scenarios, with its obvious implications for what today is described as the technology and standards for the Semantic Web. We believe Pepper solves some, if not most, of these challenges elegantly and we would like to raise this issue as one that appears very important for discussion at the workshop.

6. References

[1] R-Objects Pepper. Available on-line at http://www.r-objects.com/
[2] Object Management Group: Unified Modeling Language. Available on-line at http://www.omg.org/
[3] Java and JavaBeans. See http://java.sun.com/
[4] Project JXTA. On-line at http://www.jxta.org/
[5] XML Schema and RDF Schema. On-line at http://www.w3.org/
[6] Semantic Web Business SIG. On-line at http://business.semanticweb.org/
[7] Kunz and Rittel: Issues as Elements of Information Systems, Working Paper No 131, University of California, Berkeley, 1970.
[8] Ernst, J.: Is Peer Computing Real? Invited Talk, SCI 2001, July 2001.
[9] Ernst, J.: Peer-to-Peer Infrastructure Supporting the Semantic Web. International Semantic Web Working Symposium, Stanford, July 2001.

This page intentionally left blank

Author Index

Aldred, L. 50
Aronson, A.R. 157
Assini, P. 173
Behrens, C. 69
Bhattacharyya, P. 3
Chen, Q. 93
Cheng, Z. 117
De Roure, D. 131
Dumas, M. 50
Ernst, J. 184
Goble, C. 131
Hammond, B. 29
Kashyap, V. v,69
Khan, L. 93
Kochut, K. 29
Lassila, O. vii
Rindflesch, T.C. 157
Sheth, A. 29
Shklar, L. v
Singh, M.P. 117
ter Hofstede, A.H.M. 50
Unny, N. 3
Vouk, M.A. 117
Wang, L. 93