New Millennium or New Dark Ages?
ADCS99 Keynote Address

Roger Clarke
Principal, Xamax Consultancy Pty Ltd, Canberra
Visiting Fellow, Department of Computer Science,
Australian National University

November 6, 1999

See ADCS'99 web-site for link to slides of presentation

Abstract

At the end of the twentieth century, documents of all kinds are available to an extent, with a degree of convenience, and for costs so low, that our forebears would be unable to believe our good fortune.

There's a common presumption that the Internet is "bringing in the millennium", and is ensuring that we achieve and sustain openness, the end of inequities in the distribution of information, democracy, and human self-fulfilment.

Any such conclusion would be premature. The digital era has ambushed and beguiled us all. Its first-order impacts are being assimilated, but its second-order implications are not. Powerful institutions perceive their interests to be severely threatened by the last decade of technological change and by the shape of the "information economy" and "information society". During the coming decade, we will see a fightback by those institutions, who will implement technological countermeasures, and demand and gain changes to the law.

About the Speaker

Roger Clarke is a consultant in the management of information and information technology. He works through his own company, and in conjunction with the leading electronic commerce consultancy ETC - Electronic Trading Concepts Pty Ltd. He has particular expertise in electronic commerce, information infrastructure, and privacy and dataveillance. His work encompasses corporate strategy, government policy and public advocacy.

He holds degrees in Information Systems from UNSW, and a doctorate from the ANU. He was made an ACS Fellow in 1985, and awarded a ComputerWorld Fellowship and an IFIP Outstanding Service Award, both in 1992. In April 1996, and again in April 1997, he was named by Information Age magazine as one of the 50 most influential people in IT&T in Australia.

He has spent 30 years in the IT industry, as professional, manager, consultant and academic. This included more than a decade as a senior information systems academic at the Australian National University. He continues as a Visiting Fellow in the ANU's Department of Computer Science.

Some of the particular areas in which he has been recently active include electronic commerce policy and strategy; smart card policy and strategy; electronic payment mechanisms; electronic publishing; information infrastructure policy; Intranet and Extranet strategies; privacy strategy for corporations and government agencies; and specific technological threats to information privacy.

He has been an active participant in Internet communities throughout the 1990s, through seminars, conference papers, *e*-lists and a substantial set of community-service web-pages, including the world's most authoritative pages on *Waltzing Matilda*.

Reader's Preferences in the Formats of Web-based Academic Articles

Y. Rho
School of Computer Sci. & Eng.
University of New South Wales
yrho@cse.unsw.edu.au

T.D. Gedeon
School of Information Tech.
Murdoch University
tdgedeon@murdoch.edu.au

C.K. Kim
Department of Computer Sci.
University of Incheon, Korea
ckkim@lion.incheon.ac.kr

Abstract

No standard format exists for the many academic articles available on the Web and little is known about user reading patterns. This paper explores these issues using data from two online surveys: one email-based, the other Web-based. Our results suggests that people take an overview from the screen, and then, if they are interested in an article, print it out in order to read it properly. The simple two-frame format was regarded as the best by 47% of the respondents, whereas the cascade page format was regarded as the worst by 65% of the respondents. Interestingly, 26% considered the paper-like format, widely used in Web-based articles, to be the worst. Different results were obtained when interactive examples were embedded in the survey.

Keywords: Web-based articles; Reading patterns and formats; Web-based survey; Digital library

1. Introduction

A number of Web sites exist that publish academic articles. There are no well-developed formatting guidelines for these sites and so a variety of different formats are in use. This paper presents survey information which should be useful in determining how best to format these sites.

Articles on the Web

One format being widely used on the Web is shown in Figure 1. This format is very similar to its paper-based counterpart. We chose some well-known sites in the HCI area: conference sites such as WWW6, WWW8 and ACM CHI97, and journal and magazine sites such as ACM SIGCHI Bulletin, International Journal of Human-Computer Studies (IJHCS), Alertbox and ACM Transactions on Computer Human Interaction (TOCHI). shows some features of article formats presented on those sites. These features can be summarised as follows:

- Abstracts are always in the articles; tables of contents (TOC) are sometimes included;
- Scrolling is a common method of navigation;
- If indexing based on a TOC exists, sliding is used as a secondary method at some sites;
- The single window layout is most popular; and
- The ACM TOCHI seems to use the Web only as a delivery medium.

The formats of those sites are different as described. This may be due to the designers' different assumptions on user reading patterns. Two online surveys were carried out to investigate reading patterns and formats. Our survey participants were volunteer researchers in information technology and related areas.

Table 1 : Various formats of legacy Web-based articles

	Information overviews	Manipulation methods	Windows layout	Others
WWW6 [17]	Abstract only	Scrolling	Single window	
WWW8 [18]	• Abstract only or • Abstract & TOC or • TOC & Abstract	• Scrolling only or • Sliding (Prev-Next) or • Indexing + Scrolling	Single window	Major format is paper-like.
CHI97 [10]	Abstract – TOC	Indexing + Scrolling	Single window	
ACM SIGCHI Bulletin [11]	TOC – Abstract	• Scrolling	Single window	
IJHCS [14]	• Abstract and TOC in frames • Abstract only in single window	• Downloading to print • (Indexing + sliding) frames • Scrolling	• Single window • Multiple frames	• Multiple formats available • Comments
Alertbox [13]	Abstract only	Scrolling only	Single window	Path on the top
ACM TOCHI [12]	Abstract and general terms	Downloading to print	Single window	Delivery purpose

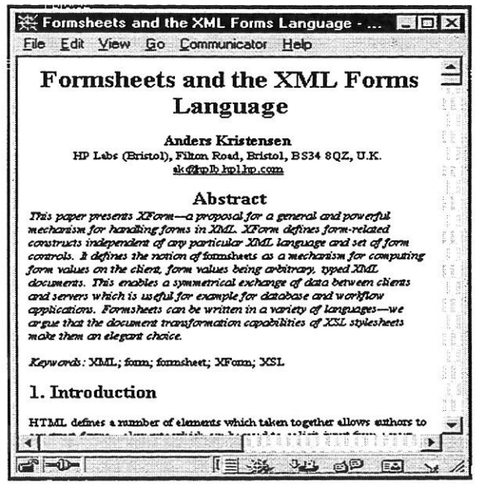


Figure 1: A popular format [15]

Online survey

The online survey (see [7][5]) is considered to be at least as good as the paper and pencil survey. In fact, it is better because it encourages participants to give more comments [9]. In the case of Web-based applications the benefit of being online is that interactive examples can be embedded in a questionnaire. These examples can provide participants with a more immediate experience of the targets in question. The use of this method is not reported as yet.

2. Email-based survey on reading patterns: the first survey

Purposes of the survey

The purpose of this survey was to determine whether researchers find research articles from the Web. If they do, what are their usage patterns?

Method

An Email-based questionnaire of four questions was distributed to 130 research people in the School of Computer Science and Engineering of the University of New South Wales in Australia. No examples were included in the questionnaire (Figure 2).

Results

We received 23 replies: that is, 18% of those 130 polled. 22 (96%) indicated that they find articles from the Web.

```
Q1: Do you find academic articles from the
    Web? (Yes / No)

If 'Yes' for Q1,
Q2: What describes your behaviour best when
you have an article on the Web?
(1) You just print it out, and then read
    the printed article.
(2) You read the first few lines on the
    first screen, print out the article if
    you are interested in it, and read the
    printed article.
(3) You read some concise parts such as
    titles & abstracts, print out the
    article if you are interested in it,
    and read the printed article.
(4) You scan through the article, print it
    out if you are interested in it, and
    then read the printed article.
(5) You read the article from the screen.
(6) Others (please describe)

Q3: What could be your second choice in Q2?

If 'No' for Q1,
Q4: Why?

    (1) The Web articles are NOT credible.
    (2) The Web articles are frequently
        updated.
    (3) Others (please describe)
```

Figure 2: Questions in the email questionnaire

Table 2: usage patterns with Web articles

	Details	1st (Q2)		2nd(Q3)	
		Frequency		Frequency	
Pattern 1	Print and read	1	5%	1	5%
Pattern 2	Read from the first screen, print and then read	0	0%	6	30%
Pattern 3	Read concise parts, print and then read	14	64%	6	30%
Pattern 4	Scan through, print and then read	7	32%	4	20%
Pattern 5	Read from the screen	0	0%	3	15%
Others		0	0%	0	0%
N		22	100%	20	100%

Table 2 shows the responses regarding usage patterns. For Q2 in Figure 2, 64% had Pattern 3 as their first choice and 32% had Pattern 4 as their first choice. The other responses were not significant. An interesting result is that no one selected Pattern 2 as their first choice. For the second choice, Patterns 2 and 3 together recorded the highest selection frequency with six (30%).

Discussion

The most common usage pattern for Web-based articles is Pattern 3. However, Pattern 4 should not be ignored as it is the first choice for 32%. These responses are very similar to the results regarding usage patterns of paper-based academic journals shown in [1]. This may be due to the fact that the reader's reading behaviour is guided by the metastructure they are used to (see [2][3] for metastructure). This metastructure seems to guide

article readers to the reading process shown in Figure 3. The respondents who prefer Pattern 3 appear to want to view the concise parts on the first screen. On the other hand, the respondents who prefer Pattern 4 seem to focus much more on the overall contents of an article than on its interface.

Survey conclusion

Those readers who find articles on the Web mostly take an overview from the screen, after which they print them out and read them. They seldom read entire articles from the screen. So which formats then do readers prefer and which do they dislike? The second survey answers this question.

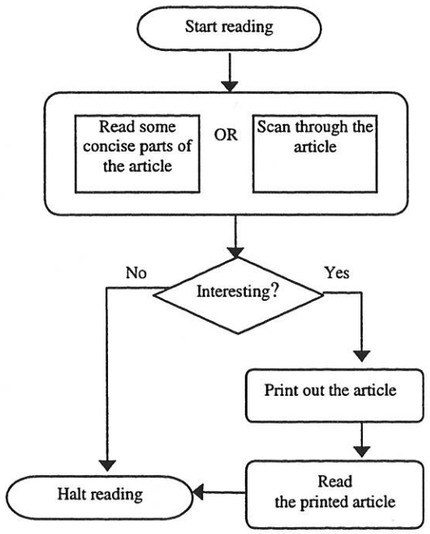


Figure 3: A reading pattern from the 1st survey

3. Web-based survey on reading patterns and interface formats: the second survey

The previous survey revealed the common reading pattern. A Web-based survey, based on this knowledge, was also conducted. Examples were provided to the participants in a Web-based questionnaire, consisting of checklist style questions. Figure 6 shows a partial screen shot of the Web-based questionnaire used in the survey.

Purposes of the survey

The second survey had two purposes. One was to identify which format readers prefer and how they use it. The other was to investigate the effects of interactive examples in the questionnaire.

Methods

A Web-based questionnaire consisting of 18 questions (i.e., [6]) was developed. The first part of this questionnaire concerns the environment settings. This part controls the visual properties and the window size of a browser in terms of the amount of information amount. This helps to avoid visual volume effects when presenting interactive examples.

The second part consists of questions about three different layers: overview types, window layouts and manipulation methods. The data from this part is not discussed in this paper. The last part is about usage patterns and overall preferences in interface formats. Each question has at least one corresponding example link.

The URL for the survey questionnaire was sent by email to 150 researchers in information technology and related areas. They were research students, research staff and academic staff. Undergraduate and coursework students were not included because they seldom use the Web to find academic articles.

Results

We received 34 replies; i.e. 23% replied to the questionnaire. Most of them used 17" monitors to complete the questionnaire (Mean=17.1").

Table 3: Reading patterns

	Details	1st choice		2nd choice	
Pattern 1	Print and read	2	6%	2	6%
Pattern 2	Read from the first screen, Print and then read	6	18%	10	29%
Pattern 3	Read concise parts, Print and then read	19	56%	8	24%
Pattern 4	Scan through, Print and then read	6	18%	10	29%
Pattern 5	Read from the screen	1	3%	4	12%
Others		0	0%	0	0%
N		34	100%	34	100%

Like the previous email-based survey, Table 3 shows that Pattern 3 (at 56%) is the most common of the reading patterns. Patterns 2 and 4 (both 18%) were the second most common. In the previous survey, there was 0% for Pattern 2 and 32% for Pattern 4. Obviously, there is a large discrepancy between the results of the two surveys. The possible reason for this difference will be discussed in the Discussion section

Table 4: Preferences in overall formats

		Best		2nd		3rd		4th		Worst	
		Freq.		Freq.		Freq.		Freq.		Freq.	
1	Paper-like	2	6%	7	21%	9	26	6	18	9	26%
2	Paper-like with TOC	12	35%	13	38%	5	15	4	12	0	0%
3	Two frames	16	47%	5	15%	7	21	5	15	0	0%
4	Slides	2	6%	8	24%	10	29	12	35	1	3%
5	Cascades	2	6%	1	3%	2	6	5	15	22	65%
	N/A	0	0%	0	0%	1	3	2	6	2	6%
	N	34		34		34		34		34	

Table 5: Correlation between reading patterns and formats for the two first choices

	Format 1			Format 2			Format 3			Format 4			Format 5			N(pattern)	
Pattern 1	0			2	100%	17%	0	0%	0%	0			0			2	6%
Pattern 2	1	17%	50%	1	17%	8%	3	50%	19%	0			1	17%	50%	6	18%
Pattern 3	1	5%	50%	6	32%	50%	9	47%	56%	2	11%	100%	1	5%	50%	19	56%
Pattern 4	0			2	33%	17%	4	67%	25%	0			0			6	18%
Pattern 5	0			1	100%	8%	0		0%	0			0			1	3%
N(format)	2	6%			35%		16	47%		2	6%		2	6%		34	100%

Table 4 shows that 47% of the respondents chose the two-frame format out of the five example formats (try Q5 in [6] for the examples). 35% selected the paper-like-with-TOC format as the best. 65% selected the cascades as the worst. It is interesting that 26% of the respondents selected the paper-like format as the worst. It should be noted that no one indicated either the paper-like-with-TOC format or the two-frame format as the worst.

Table 5 shows a correlation between the patterns and the formats. The underscored percentages of the shaded row show the format distribution over Pattern 3. Format 3's contribution to Pattern 3 is the highest at 47% and Format 2 is next at 32%. In the other view, the shaded column shown corresponds to the pattern distribution over Format 3. Pattern 3 is at 56%, followed by Patterns 4 and 2. Pattern 3 matches Format 3 and vice versa. In addition, Patterns 2 and 4 match Format 3 as well, at 50% and 67% respectively.

Discussion

The most common reading pattern for Web-based articles is for readers to look at an article in brief on the Web, and then they print it out (if they are interested in it) and then read it. The second survey produced a similar result to the first survey.

Patterns 2 and 4 revealed a big difference between the results of the two surveys. In the first survey, Pattern 4 polled 32% and Pattern 2 polled 0%. Then, in the second survey, Pattern 4 polled only 18% (a 14% decrease) while Pattern 2 polled 18% (an 18% increase). So the portions are even. Why? There was no significant difference in the distribution list and the question for this topic.

The difference seems to relate to the presence of formatting examples. The first questionnaire was based on email with no examples to try. However, the second questionnaire was based on the Web with many examples. The participants had many chances to try different formats of Web-based articles before answering the questions. They were likely to be aware of the importance of seeing the first screen. There were also some comments on that point from some participants.

The two-frame format in Figure 4 was most preferred by the respondents, which goes against the popular idea that using frames is bad. With respect to the major reading patterns the common bridge from screen to paper is printing. Therefore, article interfaces for readers have to be able to support two different media: the Web and paper. The majority (47%) chose the two-frame format as the best for the purpose and the paper-like-with-TOC format as the next (35%). 65% chose the cascade format as the worst (65%), but no one selected either the two-frame format or the paper-like-with-TOC format as the worst. The patterns and formats showed a correlation: Pattern 2, 3 and 4 most closely match the two-frame format (Table 5).

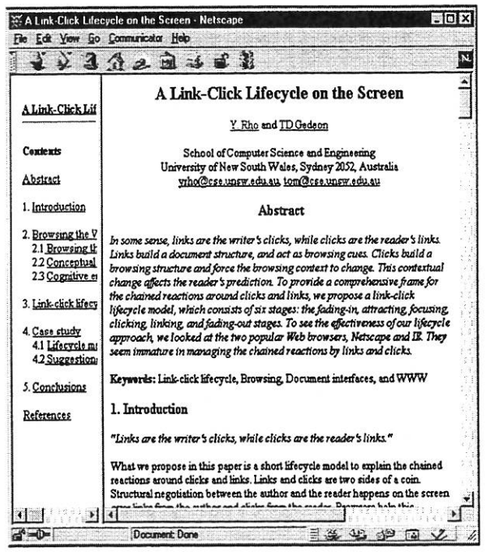


Figure 4: The two-frame format examined in the survey [16]

Survey conclusion

The early part of the user reading process, which is to take an overview of an article, happens on-screen (Figure 5). The survey results show that the early part consisting of taking an overview and printing are most likely to be supported by the two-frame format (Figure 4), which ensures the quality of the paper-based article format when printed.

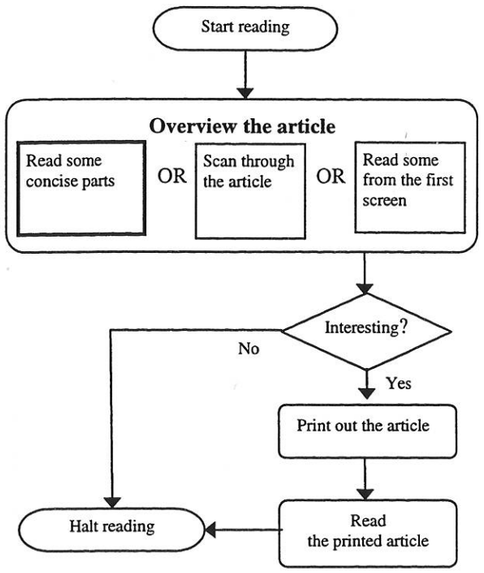


Figure 5: A typical reading process pattern with Web-based academic articles

4. Conclusion

User's reading patterns and preferences in the formats for Web-based academic articles were investigated in two online surveys: the first was email-based, the second Web-based.

The survey results show the following:

- The two-frame format (Figure 4), which consists of a TOC frame for navigation and a content frame for information, was the most favoured.
- Providing two versions together, one in the two-frame format and the other in the paper-like-with-TOC format, seems reasonable as the preference percentage (47%) for the two-frame format is not clearly dominant over the 35% of the other.
- The cascaded page format was considered by 65% to be the worst.
- An interesting discovery is that a quarter of the respondents liked least the paper-like format which is widely used.
- The use of interactive examples in the Web-based questionnaire seems to have made participants recognise the importance of the first screen.

This research focused on structural properties of Web-based academic articles from the viewpoints of information, interfaces and interactions (in short, III). Many other features including reference to other articles, multimedia presentation and annotation need to evolve to make academic articles more readable on the Web. Further research on details should be done with real reading tasks. In addition, Nielsen's claim on the inverted pyramids ([4]) needs be tested from the III viewpoint. Do we really have to write academic articles in the structure of inverted pyramids or can we provide navigation aids instead?

References

[1] Dillon, A., "New Technology and the Reading Process," Computers in Libraries, July 1991, 23-26.

[2] Dillon, A., "Readers' models of text structures: The case of academic articles," International Journal of Man-Machine Studies, 35, 1991, 913-925.

[3] Dillon, A., " Expertise and the Perception of Shape in Information," Journal of the American Society for Information Science, 47(10), 1996, 786-788.

[4] Nielson, J. "Inverted Pyramids in Cyberspace," Alertbox, 1996. http://www.useit.com/alertbox/9606.html

[5] Perlman G., "Web-Based User Interface Evaluation with Questionnaires," 1997. http://www.acm.org/~perlman/question.html

[6] Rho, Y., Questionnaire on Reader's preferences in Web-based Academic Articles, 1999. http://www.cse.unsw.edu.au/~yrho/WebPaper/Question.htm

[7] Root, R. W., Draper, S., "Questionnaires as a software evaluation tool." CHI'83 Proceedings, December, 1983, 83-87.

[8] Shneiderman, B., "Designing Information-Abundant Websites: Issues and Recommendations," International J of Human-Computer Studies, 1997. http://www.cs.umd.edu/projects/hcil/members/bshneiderman/ijhcs/main.html

[9] Slaughter, L., Harper, B., and Norman, K., "Assessing The Equivalence Of The Paper And On-line Formats Of The Quis 5.5". Proceedings of the 2nd Annual Mid-Atlantic Human Factors Conference, Washington, D.C, 1994, 87-91. http://lap.umd.edu/LAPFolder/Papers/SHN.html

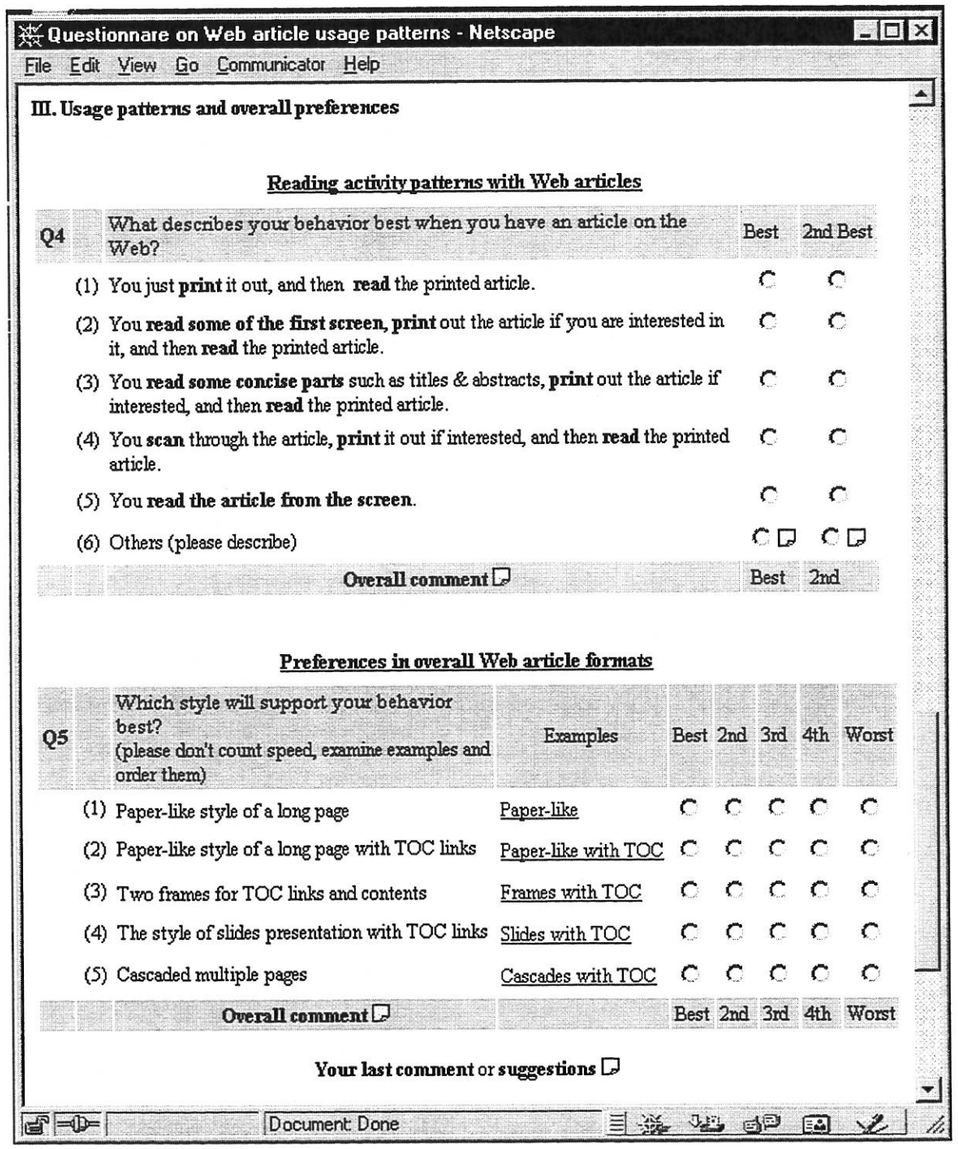
Example URLs

[10] ACM CHI97, 1997. http://www.acm.org/sigchi/chi97/proceedings/paper/plp.htm

[11] ACM/SIGCHI Bulletin, 1997. http://www.acm.org/sigchi/bulletin/1997.4/ross.html

[12] ACM Transactions on Computer-Human Interaction.

http://www.acm.org/pubs/contents/journals/tochi/1998-5/
[13] Alertbox, 1997. http://www.useit.com/papers/webwriting/writing.html
[14] International Journal of Human-Computer Studies, 1997. http://ijhcs.open.ac.uk
[15] Kristensen, A., "Formsheets and the XML Forms Language," WWW8 Proceedings, 1999. http://www8.org/w8-papers/1c-xml/formsheets/formsheets.html
[16] Rho, Y., Gedeon, TD, "A Link-Click Lifecycle on the screen," APWeb'98 Proceedings, Beijing, 1998. http://www.cse.unsw.edu.au/~yrho/Publications/apweb98/apweb98_C-tiled_main.htm
[17] WWW6 Conference, 1996. http://www.scope.gmd.de/info/www6/technical/paper003/paper3.html
[18] WWW8 Conference, 1999. http://www8.org/fullpaper.html


Questionnare on Web article usage patterns - Netscape

File Edit View Go Communicator Help

III. Usage patterns and overall preferences

Reading activity patterns with Web articles

Q4	What describes your behavior best when you have an article on the Web?	Best	2nd Best
(1)	You just **print** it out, and then **read** the printed article.	○	○
(2)	You **read some of the first screen, print** out the article if you are interested in it, and then **read** the printed article.	○	○
(3)	You **read some concise parts** such as titles & abstracts, **print** out the article if interested, and then **read** the printed article.	○	○
(4)	You **scan** through the article, **print** it out if interested, and then **read** the printed article.	○	○
(5)	You **read the article from the screen**.	○	○
(6)	Others (please describe)	○	○
	Overall comment	Best	2nd

Preferences in overall Web article formats

Q5	Which style will support your behavior best? (please don't count speed, examine examples and order them)	Examples	Best	2nd	3rd	4th	Worst
(1)	Paper-like style of a long page	Paper-like	○	○	○	○	○
(2)	Paper-like style of a long page with TOC links	Paper-like with TOC	○	○	○	○	○
(3)	Two frames for TOC links and contents	Frames with TOC	○	○	○	○	○
(4)	The style of slides presentation with TOC links	Slides with TOC	○	○	○	○	○
(5)	Cascaded multiple pages	Cascades with TOC	○	○	○	○	○
	Overall comment		Best	2nd	3rd	4th	Worst

Your last comment or **suggestions**

Document: Done

Figure 6: A partial screen shot of the Web-based questionnaire [6]

DYNAMIC HYPER-LINKING BY QUERYING FOR A FCA-BASED QUERY SYSTEM

Bernd Groh
School of Information Technology
Griffith University
PMB 50 Gold Coast Mail Centre
QLD 9726, Australia
b.groh@gu.edu.au

Peter Eklund
School of Information Technology
Griffith University
PMB 50 Gold Coast Mail Centre
QLD 9726, Australia
p.eklund@gu.edu.au

Abstract

This paper presents a mechanism for hyper-linking documents by search-terms. Search-terms are selected by the user interactively building a formal concept lattice. In order to explain this interface we give some background to Formal Concept Analysis and an example is developed which illustrates the use of the concept lattice. Selected search-terms are used to create hyper-links, based on term repetition.

As the search-terms differ between queries, we need a mechanism from which to dynamically create the target hyper-linked HTML documents. Therefore, documents are stored in a structure which is based on a word-list rather than plain text format. The documents are represented as links between the individual words within the word-list. In so doing the word-list becomes a full-text-retrieval index into each word in each of those documents and therefore provides a good basis for the fast creation of an HTML document set from specific queries by keywords.

To have the words in a word-list from which the documents are created also allows easy classification of words which should be hyper-linked within specific HTML documents. Furthermore, both documents and hyper-linking keywords are stored as well in this structure since any word in any document is indexed by the word-list.

Keywords: Document Databases, WWW and Internet.

1 Introduction

There have been several developments in automatically generating hyper-linked documents, from hyper-linking by term repetition to semantic approaches based on similarity measures between documents. One approach by Green [7, 8] uses lexical chains [9] to measure the similarity between documents. This mechanism is based on WordNet [5] and generates groups of related words within documents. Those groups have specific meanings and can be related to groups with equivalent or similar meanings in other documents.

Our approach is query-driven and focuses on keywords. We use concept lattices as a visual interface so that the user can navigate to a specific subset of documents with given combinations of search-terms (or keywords) of interest. The hyper-links will be created based on repetition of those terms and will exist only between documents in the specific subset that satisfies the query. The technique to create the concept lattices is called Formal Concept Analysis (FCA) [6].

We structure the paper with a brief introduction to FCA. Next, based on an example of a medical document-set, we demonstrate how to read a concept lattice and how it helps the user to identify documents of interest. As document-sets differ, depending on the terms appearing in the concept lattice and on the selected concepts within the lattice, hyper-links have to be created dynamically.

In Section 4, we present a mechanism to dynamically generate a hyper-linked document-set, wrt to a specific query, based on term repetition. The approach uses a memory structure in which the entire document collection is stored and an algorithm to create the hyper-linked document-subset. The structure is based on a word-list, containing entire words and links between the words. These links, when followed, constitute the document-set.

As our interest lies in complete words which precisely match a pre-defined list of terms, in our example medical terms, fast string searching algorithms based on substrings, such as Suffix Trees, cannot be used. Our approach is explained on a small example in Section 5 and the algorithm is described in Section 6. The last sections conclude the paper and provide an outline of current and future work.

Proceedings of the Fourth Australasian Document Computing Symposium, Coffs Harbour, Australia, December 3, 1999.

2 Formal Concept Analysis

Wille introduced in [10] the metaphor of "Landscape of Knowledge", to describe the processes by which knowledge is explored using formal concept analysis. The processes of knowledge exploration is seen as a dynamic one, in which the computer is used as a medium through which aspects of the knowledge can be investigated.

Central to the idea of formal concept analysis is the understanding that a fundamental unit of thought is a concept. The concept is constituted by its intention and its extension. This understanding had been formalised by starting with a *(formal)* context, $\mathbb{K}$ defined by a triple (G, M, I) where G and M are sets and I is a binary relation between G and M (i.e. $I \subseteq G \times M$). gIm is read as *"the object g has the attribute m"*. The *(formal)* concepts of $\mathbb{K}$ are the pairs (A, B) with $A \subseteq G$ and $B \subseteq M$ such that (A, B) is maximal with respect to the property $A \times B \subseteq I$. The set A is called the extent and the set B is called the intent of the concept (A, B). The set "under" $\mathcal{B}(\mathbb{K})$ of all concepts of a context $\mathcal{K}$ with order $(A_1, B_1) \leq (A_2, B_2) :\leftrightarrow A_1 \subseteq A_2$ is always a complete lattice, and is called the *concept lattice* of the context $\mathbb{K}$.

In our example of a medical document collection, G is the set of all documents in the collection and M is the set of all MeSH-terms[1] within the MeSH-hierarchy that appear in at least one of the documents. gIm is therefore read *"the document g contains the MeSH-term m"*.

The process of creating a concept lattice must be performed using expert knowledge from the domain from which the data is taken. Often concept lattices are created by hand. This includes the selection of terms to form a lattice and the layout of the lattice. In work by Cole [3, 4, 2] concept lattices are created by the manipulation of a view of a taxonomy of terms, i.e. the medical taxonomy of the MeSH-hierarchy. Manipulation of the view both defines the concept lattice and conditions its layout. This project was the motivation for hyperlinking documents using a concept lattice as the basis for a query interface.

3 Concept Lattices Over Medical Documents

Figure 1 shows a concept lattice for some given attributes out of a medical taxonomy, the MeSH-hierarchy. The attributes or keywords within a document are placed as labels above a node, which represents a concept in the formal context. Documents are placed as labels, containing the number of documents with those specific attributes, or keywords, below the node. The

[1]MeSH is the Medical subjects Headings [1].

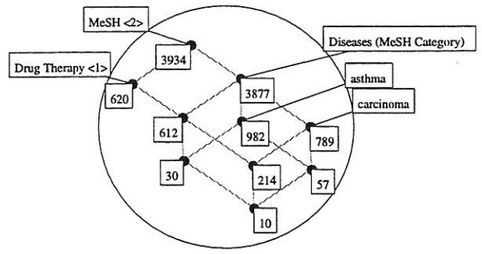


Figure 1: Lattice diagram.

lattice is best read bottom up. The node with 214 documents associated to it, for example, has the attributes "carcinoma", "Drug Therapy <1>"[2], "Diseases (MeSH Category)" and "MeSH <2>". As the documents are patient records, this means that 214 patients have "carcinoma", which falls under "Diseases (MeSH Category)" and is a term of the MeSH-Hierarchy, classified by "MeSH <2>" and are undergoing a "Drug Therapy <1>", which is also a term of the MeSH-Hierarchy. The node with 789 documents associated with it, has all those attributes, except "Drug Therapy <1>". This, in conjunction with the node with 214 documents, can be read as 789 patients do have "carcinoma" of which 214 are undergoing a "Drug Therapy <1>". The node with 10 documents also has the attribute "asthma", which can be read as 214 patients who have "carcinoma" and are undergoing a "Drug Therapy <1>" also have "asthma". 57 patients have "carcinoma" and "asthma" and are not undergoing a "Drug Therapy <1>".

The way in which the attributes are selected is done in an interactive way by the user, where she/he can select her/his area of interest, see Cole [3, 4, 2]. We are interested in the document sets on specific nodes. Each node, for every possible attribute combination, contains a specific number of documents, containing exactly those attributes or keywords.

4 Dynamic Hyper-linking Medical Documents

The process by which MeSH-terms are selected by the user is a metaphor for a query theme. This theme can then be used as the basis for generating a concept lattice showing how the documents distribute across combinations of terms in the theme. It would be useful, on the basis of creating a theme

[2]Suffixes such as <1> on a MeSH-term indicate that the term has several occurrences, in this case <1> indicates that this is the first occurrence in the MeSH-hierarchy for this term.

(a conceptual scale in the FCA literature), if we could use the theme to dynamically hyper-link the document collection. The mechanism for hyperlinking the documents has been applied to the same document collection as mentioned above. An example document is shown in Figure 2.

```
{{ Problem List }
 { 0. Small cell ca left lung, 8 cycles chemotherapy 1990, plus
   radiotherapy 1991. 1. Pulmonary embolus. 2. Glaucoma.
   3. Peptic ulcer. 4. Cholecystectomy. 5. Appendicectomy.
   6. Oophorectomy. 7. Right sided pneumonia and neutropaenia. }}
{{ Discharge Treatment }
 { Coloxyl with senna 2 tabs bd, Ventolin 90 sec 4 hrly prn,
   Ranitidine 300mg bd, Mylanta 20 mls tds prn,
   Nifedipine 10mg tds, Panadeine forte 1-2 tabs 4 hrly prn,
   Dipiverine Hydrochloride .1% 2 dps bd both eyes. }}
{{ Information to Patient }
 { Patient aware of diagnosis and limited prognosis.
   Knows to present to LMO or RAH with any problems. }}
{{ Summary of Admission }
 { 68 year old woman, well known to S2. Discharged one week ago.
   Day after discharge, developed increasing SOB with
   yellow/white sputum production. Felt unwell, but denies
   rigors or chills. Using Ventolin regularly with no
   improvement. Transferred by ambulance to RAH. }}
{{ Examination }
 { ; mildly tachypnoeac, RR 30, not cyanosed, looks unwell,
   febrile 38.9, dehydrated, HR 120/reg, BP 140/90, JVP NR,
   HS dual + nil, no ankle swelling, peripheral pulses all
   present, TKL, PN dull left base, BS vesicular, reduced
   at left base, crackles right anterior chest. Abdominal
   and neurological examination unremarkable. }}
{{ Investigation }
 { }}
{{ Progress }
 { a steady improvement made. Freely mobile around the ward
   without oxygen on discharge. }}
{{ Follow Up }
 { Chest Clinic/Dr. Holmes LMO to perform MBA20 prior to OPD
   appt. }}
{{ Copies }
 { lmo,file,ur. }}
```

Figure 2: Medical document as used.

If we are interested in documents that contain both the terms "carcinoma" and "Drug Therapy <1>" for example, we select the node in Figure 1 with 214 documents. The first document in this set of 214 is shown in Figure 3.

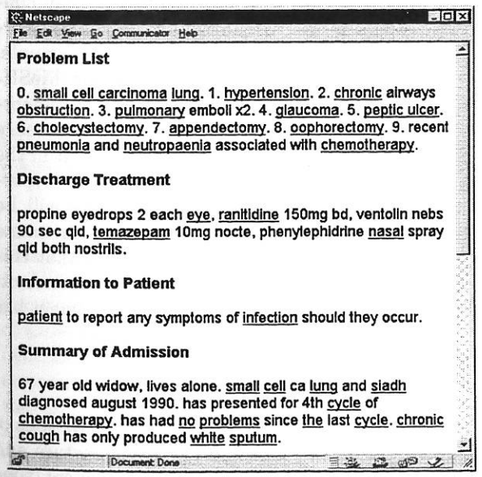


Figure 3: First of 214 generated HTML documents.

Every term that is in the hierarchy (in the example the MeSH-Hierarchy) is either linked to the next document in the retrieved document set, containing that term, or is boldfaced to indicate there is no other document in the retrieved set that contains that term. In Figure 3, all terms that are contained by at least one other document in the retrieved set are linked to the next document with that specific term occurrence. The last occurrence of that term in the last document, containing it, refers back to the first occurrence of that term. The search terms "carcinoma" and "Drug Therapy <1>" appear in every document and it is therefore possible to view every document in the retrieved set by following those links. The problem is then to produce a fast and efficient way to generate these hyper-linked HTML documents.

5 Efficiently Representing Documents

As the retrieved document sets differ each time, depending on the query (or the conceptual scale) and the hyper-links within the documents differ each time, it is not desirable to create all possible HTML documents in advance. Therefore, the documents have to be stored in a way that is as easy and efficient as possible to create specific HTML documents on demand. As we work with text documents, where attributes are words (or phrases), the easiest thing is to store documents as linked lists of words. The first thing to do is to create a list of all the words in all the documents — the index. Given we have the following three documents:

```
The house is blue.

My house is nice.

My pen is blue.
```

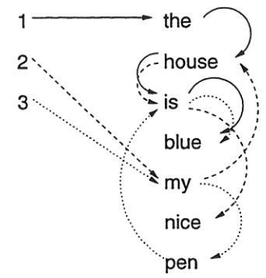


Figure 4: Linked list of the 3 documents.

Figure 4 shows the representation of the linked lists for the three documents. How should this be stored physically?

DOCUMENT	ADDRESS NEXT WORD
1	&the
2	&my
3	&my

WORD	CLASSIFICATION	WORDS LINKED	ADDRESS NEXT WORD	NUMBER LINKS	DOCUMENT ID	WORD ID	DOCUMENT ID	WORD ID	ADDRESS NEXT WORD	NUMBER LINKS	DOCUMENT ID	WORD ID
the	0	1	&house	1	1	1						
house	0	1	&is	2	1	2	2	2				
is	0	2	&blue	2	1	3	2	3	&nice	1	2	3
blue	0	0										
my	0	2	&house	1	2	1			&pen	1	3	1
nice	0	0										
pen	0	0										

Figure 5: Memory structure of the linked documents.

Figure 5 shows how the data is stored in memory. The figure is a table representation for an easier understanding of the structure. Each document is stored with just the address of the first word. Document 1, for example, points to "the", which is the first word of document 1. Of course other information such as filename can be stored as well, but there is just one pointer into the document itself. For every word is stored:

1. an id which represents the classification of the word or its place in the hierarchy;
2. the number of links to a following word;
3. for each link to a following word;
   (a) the address of the next word;
   (b) the number of links to that specific word;
   (c) for each link to that specific word;
      i. the document number;
      ii. the word number.

From this data structure the documents can be reproduced. In Figure 5 document 1 points to "the". From "the", there exists just one possible link, to "house", and to word 1 in document 1 (we count "the" as 0). From "house" there is one possible link, to "is". There are two possible links and one of them is to word 2 in document 1. From "is" there are two possible links, to "blue" and to "nice". One of the links to "blue" is to word 3 in document 1, the next word therefore is "blue". From "blue" there are no further links and the end of the document is reached.

The restored words "the", "house", "is", "blue" constitute document 1. In the implementation, the difference between upper- and lowercase as well as special characters and set signs have been considered and are stored in additional entries, but we will not talk about that in this paper. As we are creating HTML documents, we are simply storing information to reproduce the text, formatting may be lost. In our testing we use plain text-documents. Nevertheless this mechanism is extendible to accommodate further text-options that can be represented by HTML.

6 Building the HTML Documents

After the user has made a selection of the document-subset of interest, each document in the subset will be regenerated from the memory structure.

Figure 6 shows the algorithm, which loops over all documents in the subset. For each document the algorithms run through the word-list, collecting all the words within the document. If the classification id is non-zero, this is a term of the hierarchy or the beginning of a phrase within the hierarchy, in our example the MeSH-hierarchy. If the classification id is zero, the word is output to the current HTML document and the algorithm can continue.

When the current word is classified, the algorithm tests whether it is a one-word term or the beginning of a phrase within the MeSH-hierarchy. Words collect, until the phrase is maximal – long phrases have higher priority than short phrases, short phrases a higher priority than single words. The maximal phrase in the MeSH-hierarchy constitutes the term. If the words "Small cell carcinoma" collected through the links, for example, "Small" will not be used, even if it is a MeSH-term. Next, the algorithm looks up the document-word pairs that contain the first word occurrence (or phrase occurrence) in the next document in the specified subset. That is the position the hyper-link is construct to. Now the hyper-link will be output into the HTML document, where the name of the hyper-link is the word-number, so that this link can be linked directly from another document. This is repeated, until the end of the document has been reached.

7 Future Work

We plan to investigate different implementations of the algorithm to create the memory structure and examine differences in memory management/time complexity issues. We also want to investigate different implementations of physically storing the links and see whether we can achieve better memory management without changes to time

Inputs
Let G be the array of the document collection
Let AID be a set of document ids
Let B be a set of MeSH-terms
Outputs
Let HTA be an array of HTML documents
Variables
Let *next* be the address of a word within the word-list
Let *term* be a MeSH-term
Let *dn* be the id of a document
Let *wn* be the position of a word within a document
Let *wc* be the counter of the words within the current document
Algorithm

```
FOR EACH (aid ∈ AID)
  PrintHeader(hta[aid], g[aid])
  wc = 0
  next = g[aid].FirstAddress
  WHILE (next)
    IF (next →Classification = 0) THEN
      Print(hta[aid], next →Word)
    ELSE
      GetMeshTerm(term, next, wc)
      IF (GetNextOccurrence(dn, wn, term) = 0)
        PrintNoLink(hta[aid], term)
      ELSE
        IF (term ∈ B) THEN
          PrintActiveLink(hta[aid],
                wc, dn, wn, term)
        ELSE
          PrintPassiveLink(hta[aid],
                wc, dn, wn, term)
        END IF
      END IF
    END IF
    next = GetNextWord(aid, wc)
  END WHILE
  PrintFooter(hta[aid])
END FOR
```

Figure 6: Algorithm to create the HTML documents from the memory structure.

complexity. In doing so, we will also consider compression techniques.

What takes up most memory are the document/word pairs and the average/max length of those depend on the number of documents in the collection and the number of words within a single document. Given different cases we will compare the presented structure with other techniques, inverted file indexing for example, to see how it compares with other techniques we could use for this purpose. Finally, we want to build a tool that is general enough for most document-sets. It should automatically create the memory structure out of the document collection, allow to plug-in any hierarchy, on which the hyperlinks are based, and that then creates the HTML documents from any query.

8 Conclusion

A mechanism has been presented that allows a user to define terms (attributes) and use them to generate hyper-linked HTML documents. It has been shown that a structure to store these documents results in easy and efficient access to the documents.

Apart from fast composition of HTML documents this structure also provides a full-text retrieval index into each word in each document. Furthermore, it allows easy classification of the single words or phrases and an easy embedding of a hierarchy over those words or phrases. It also allows easy access to information about the documents, such as word-count, number of a specific link or a specific word combination. Finally, the resulting structure might require less memory than the plain documents itself. This, of course, is dependent on the document collection, but it was valid for our test-set. Once this structure has been created, it is no longer necessary to keep the original documents, as the plain text documents can be fully restored from that structure.

To compare the memory needs: The 4,000 documents in our test-set took up storage space of 7.23 MB. The newly created structure consumes 5.12 MB. Given that reduction of storage space was not the aim of this work it is nevertheless an appreciated side-effect. Just to mention the compression results we got using gzip^{TM}. The Zip file of the 4,000 documents, using maximum compression, has a size of 4.13 MB, the size of the Zip file of our structure is 3.88 MB. Of course, it would be possible to use further compression techniques, but our aim was fast creation of HTML documents and we therefore have decided not to use any. In the current version we work with full byte-lengths, even when we need only a few bits because access to full bytes is easier to implement. With most compression techniques it would be difficult to apply techniques, such as binary search, to reduce the time for creation of HTML documents.

References

[1] *1998 MeSH, Annotated Alphabetic List.* National Technical Information Service, U.S. Department of Commerce, Springfield, VA 22161, 1998.

[2] Richard Cole and Peter Eklund. Scalability in formal concept analysis. *Computational Intelligence: An International Journal*, Volume 15, Number 1, pages 11–27, February 1999.

[3] Richard Cole, Peter Eklund and Bernd Groh. Dealing with large contexts in formal concept analysis. In *Second International Symposium on Knowledge Retrieval, Use and Storage for Efficiency*, pages 151–164, Vancouver, B.C., Canada, August 1997.

[4] Richard Cole, Peter Eklund and Don Walker. Using conceptual scaling in formal concept analysis for knowledge and data discovery in medical texts. In *Second Pacific Asian Conference on Knowledge Discovery and Data Mining*, 1998.

[5] Christiane Fellbaum (editor). *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[6] Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer-Verlag, 1999.

[7] Stephen Green. *Automatically generating hypertext by computing semantic similarity*. University of Toronto, Canada, 1997.

[8] Stephen Green. Automated link generation: Can we do better than term repitition? In *Proceedings of the Seventh International World Wide Web Conference*, pages 75–84, Brisbane, Australia, April 1998.

[9] Jane Morris and Graeme Hirst. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, Volume 17, Number 1, pages 21–48, 1991.

[10] Rudolf Wille. Landscapes of knowledge: A pragmatic paradigm for knowledge processing. In *Second International Symposium on Knowledge Retrieval, Use and Storage for Efficiency*, pages 2–13, Vancouver, B.C., Canada, August 1997.

TML: A Thesaural Markup Language

Maria Lee
Mathematical & Information Sciences
CSIRO
Locked Bag 17, North Ryde 1670
Australia
Maria.Lee@cmis.csiro.au

Stewart Baillie
Mathematical & Information Sciences
CSIRO
723 Swanston Street, Carlton 3053
Australia
Stewart.Baillie@cmis.csiro.au

Jon Dell'Oro
Mathematical & Information Sciences
CSIRO
723 Swanston Street, Carlton 3053
Australia
Jon.Delloro@cmis.csiro.au

1 Abstract

Thesauri are used to provide controlled vocabularies for resource classification. Their use can greatly assist document discovery because thesauri mandate a consistent shared terminology for describing documents. A particular thesaurus classifies documents according to an information community's needs. As a result, there are many different thesaural schemas. This has led to a proliferation of schema-specific thesaural systems. In our research, we exploit schematic regularities to design a generic thesaural ontology and specify it as a markup language. The language provides a common representational framework in which to encode the idiosyncrasies of specific thesauri. This approach has several advantages: it offers consistent syntax and semantics in which to express thesauri; it allows general purpose thesaural applications to leverage many thesauri; and it supports a single thesaural user interface by which information communities can consistently organise, store and retrieve electronic documents.

Keywords: Electronic Documents, Metadata, Ontology, Thesaurus, XML

2 Introduction

Many problems common to electronic document systems are often not new, but well-known problems occurring in a new medium. In a search for solutions to problems in the electronic medium, we can often learn from the experience of traditional media. This is true for resource discovery in large electronic information repositories. The solutions offered by search engines have evolved rapidly to fill a need for resource discovery in the electronic storage medium. But, in a managed information environment, their free text search approach can be a poor substitute to thesaurally organised metadata approaches.

The use of metadata search can complement and enrich the text matching approach of search engines. Metadata is data which describes data. It provides a conceptual description of a resource's content, context, and function. The keywords list at the head of this paper is an example of metadata— it describes something about the document content. In document management systems, metadata is often used to index a document by describing what it is about and its catalogue detail. However, this metadata content, when used for search, can run into a similar problem to that of document content: it lacks a consistent shared vocabulary. A traditional solution to this problem is to use a thesaurus to control metadata content.

In the terminology of the record keeping community, a thesaurus is a fixed vocabulary of approved and unapproved terms, their functions and meanings, and their inter-term relationships. A thesaurus can provide *accuracy* of description through explicit classification by approved terms; *consistency* through controlled terminologies; and *efficiency* in retrieval through the use of the right terminologies [Lancaster 1972].

A thesaurus is valuable if its vocabulary acts as a *lingua franca* that reflects the culture of a user community and purposes the information repository schema. This often means that a different thesaurus is necessary for each user community. The result, in the electronic storage medium to date, has been many incompatible thesaural applications each one designed about its particular thesaurus. In our research we have sought a generic ontology in which to represent the idiosyncrasies of these many specific thesauri. This would allow a single application to work with many different thesauri. In this paper, we describe this ontology, a markup language used to express it, and introduce a general purpose Thesaural Explorer application based upon them.

3 Generic Thesaural Ontology

An ontology, in computer science, has come to denote an explicitly specified conceptualisation of part of the world. In software, an ontology is implemented as a data structure. What distinguishes the ontology from the data structure is semantics: that it talks about something in the world. An ontology provides users with a representation which is essential to effective communication and coordination.

Proceedings of the 4th Australasian Document Computing Symposium,
Coffs Harbour, Australia,
December 3, 1999.

Our goal was to design a Generic Thesaural Ontology (GTO) capable of representing many different thesauri. This would allow us to express a specific thesaurus in a common language. The way we approached this goal was to review six major existing thesauri and model their classes and relations at a higher level of abstraction. The six thesauri selected were:

- Keyword AAA Australian Government Thesaurus [Keyword AAA],
- Getty Art and Architecture Thesaurus [AAT],
- Getty Thesaurus of Geographic Names [TGN],
- Library of Congress Subject Headings [LCSH],
- OCLC Dewey Decimal Classification [OCLC],
- Medical Subject Headings [MeSH].

These thesauri were selected because they are well-known, used by different communities in different domains, represented both function-based and subject-based classification schema, and are based on the monolingual thesaural standard [ISO 2788]. In the draft of the GTO presented here, it was not our goal to represent multilingual thesauri or to map inter-thesaural links.

We tried to keep the thesaural ontology as simple as possible. In this case, we found a taxonomic graph sufficient to our purposes. The classes and relations of the GTO are shown in Figure 1.

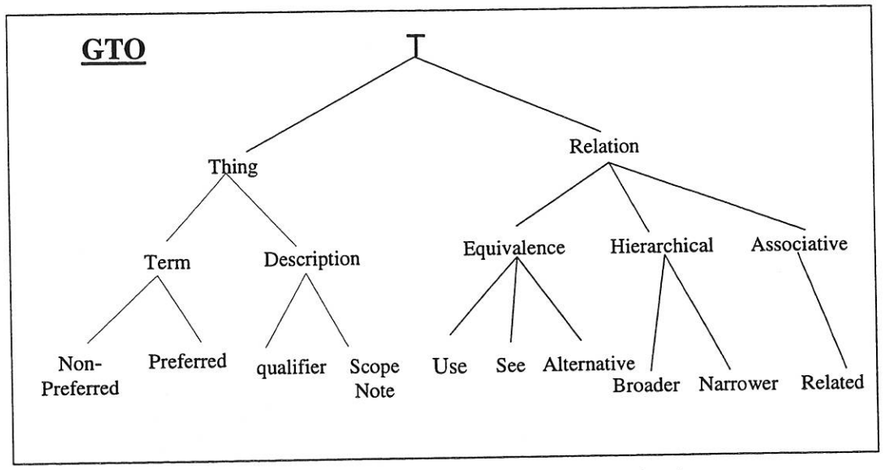


Figure 1. Graph of the Generic Thesaurus Ontology.

The GTO graph illustrates inheritance, where each class on the lower level inherits properties from the preceding level. The root symbol T is a neutral representation for the universal entity. T bifurcates to superordinate the GTO classes of *Thing* and *Relation*:

Thing:
is defined by a monadic predicate p(x) in terms of the form of the entity x (including its inherent parts and properties) and not in terms of anything external to x.

Relation:
is defined by a dyadic predicate (x,y) that relates the entity x to some independent y that is not an inherent part or property of x.

The class Thing has two subtypes:

Term:
represents any word or phrase used to represent a concept. Thesaural terms are divided into preferred terms and non-preferred terms. Preferred terms are authorised terms and the only ones valid for use in resource description. Non-preferred terms (synonyms, spelling variants, inverted form, etc) are designated by a USE relation which links them to the preferred term.

Description:
describes the meaning of a concept. It includes Scope Note description and parenthetical qualifier. A Scope Note is a statement that clarifies the meaning and usage of a descriptor or guide term within the thesaurus. The parenthetical qualifier is used to qualify or specify the context of an entry and so remove ambiguity. It allows users to distinguish among the homographs at a glance, while their scope notes further define them.

The class Relation consists of three subtypes:

Equivalence:
the equivalence relationship exists between or among terms that represents the same concept. Equivalent terms may be synonyms, variant spellings, inverted forms of multi-word terms, etc. Equivalent terms falls into three categories: Use terms, See and Alternative.

- *Use*: When a concept can be expressed by more than one term or more than one spelling, one of the terms is selected as the preferred term, and the other included as non-preferred or use-for terms. In all cases the two terms involved (referred from and referred to) are essentially equivalent. In many thesauri the use reference is also employed to effect one-to-many mapping.
- *See*: Although *use* reference is the usual thesaurus convention for directing from a term that cannot be used in indexing and searching to a term that can be used, some vocabularies prefer *see* to serve the same function.
- *Alternative*: different grammatical forms of the descriptor. Generally they are to allow for variety of indexing practices, such as use of singular instead of plural, and to provide a combination form for use in constructing headings from more than one descriptor.

Hierarchical:
is the most fundamental thesaural relationship, the basic type of links establishing a term's membership in the thesaurus. The relationship generally is restricted to the formal genus-species relation. If a term is a type of, kind of, example of, or manifestation of another term, then a genus-species relation exists. Within the context of the genus-species relationship, the genus or class is called the Broader Term and the species or member is called the Narrower Term. The broader-narrower relations are reciprocals of one another.

Associative:
relates terms that are not hierarchical (broader-narrower) nor equivalent (use) but in some other way linked. Usually they link between terms that belong to different categories, with no siblings; these provide the basis for the most common types of related terms and are the most difficult links to define rationally. Generally speaking the functions of the related terms in thesauri are to clarify the scope of and to define the main term, and to alert the indexer or searcher to other terms or concepts of interest.

4 Thesaurus Markup Language

The general thesaural ontology gives us a conceptual representation of thesauri. A thesaural markup language (TML) manifests this as a grammar in which to express the content and structure of specific thesauri. TML is specified as an XML schema which defines the permitted markup element types and embedding structure. The TML syntax consists of the element names and structure shown in Figure 2.

Figure 2. TML Element Graph

The TML graph structure is not isomorphic with the GTO graph structure; ie. it does not classify according to the thesaural ontology, but reorganises the semantic classes and relations of the GTO into a process-oriented data structure which reflects how the data are used. The TML element *thesaurus* subordinates three types of TML elements: system, ontology, and instance. The *system* element represents metadata about the thesaurus. The *ontology* element represents a particular thesaurus' structure (its idiosyncrasies); it extends the generic GTO taxonomy down to a specific thesaurus instance. The *instances* element represents the content needed to populate a thesaurus. Its prefTerm sub-element represents a preferred thesaural term. The prefTerm element is the lynchpin of the TML instance structure.

The following tables give more detail on the TML elements. In the following tables, the occurrence column indicate the existential status of each element. The meanings of the occurrence symbols are:

Occurrence Symbol	Meaning
1	Required, not repeatable
?	Optional, zero or one occurrence
+	Required, repeatable (one or more occurrence)
*	Optional, repeatable (zero or more occurrences)

The *system* element is composed of the following sub elements:

Name	Occurrence	Description
identifier	1	The name of the thesaurus
version	1	The version number of this thesaurus
language	?	The language used in this thesaurus
authority	?	Organisation authorisation
createdBy	?	The name of the person/organisation who created the record defining the term
approvedBy	?	The name of the person/organisation who approved the record defining the term
date	?	
createdDate	?	The date on which the record defining the term was created
modifiedDate	?	The date on which the record defining the term was last modified

The *ontology* element describes the thesaural GTO extensions (see Figures 4 & 5). It is composed of the following sub elements:

Name	Occurrence	Description
setPrefTerm	+	Register the type and name of a class of preferred terms
setNonPrefTerm	*	Register the type and name of a class of non preferred terms
setScopeDesc	*	Register the type and name of a class of scope notes
setQualDesc	*	Register the type and name of a class of qualifiers
SetUserRelation	*	A user defined relation

The *instances* element is composed of the following sub elements:

Name	Occurrence	Description
prefTerm	1	The instance of the preferred term
scope	?	The instance of the scope note
qualifier	?	The instance of the qualifier
broader	*	The instance of the broader term
narrower	*	The instance of the narrower term
related	*	The instance of the related term
nonPrefTerm	*	The instance of non-preferred term
userRelation	?	The instance of user defined relation

An convenient way to understand how TML works is to look at some worked examples. These are described below.

4.1 TML for Keyword AAA Thesaurus

Keyword AAA [Keyword AAA] is the thesaurus most extensively used by Australian Government agencies. It uses the relationships broader term, narrower term and related term. The broader and narrower term relations are reciprocal and the related and top relations are reflexive. Figure 3 illustrates some of these Keyword AAA terms and relations.

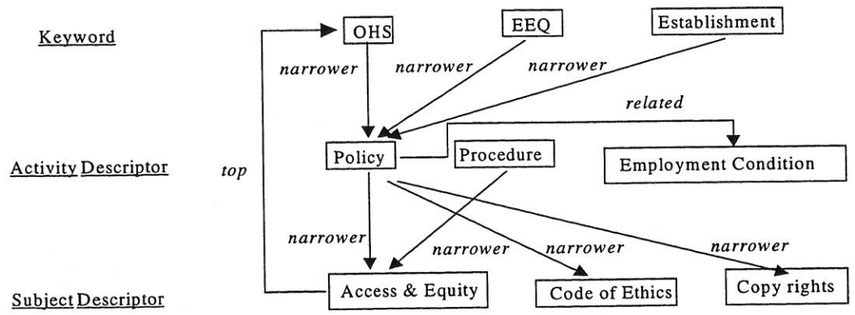


Figure 3. Keyword AAA Thesaurus Graph

The Keyword AAA thesaurus uses three types of preferred terms: Keywords, Activity Descriptor, and Subject Descriptor. Permitted Acronyms and Forbidden Terms are types of non-preferred terms. A Scope Note construct

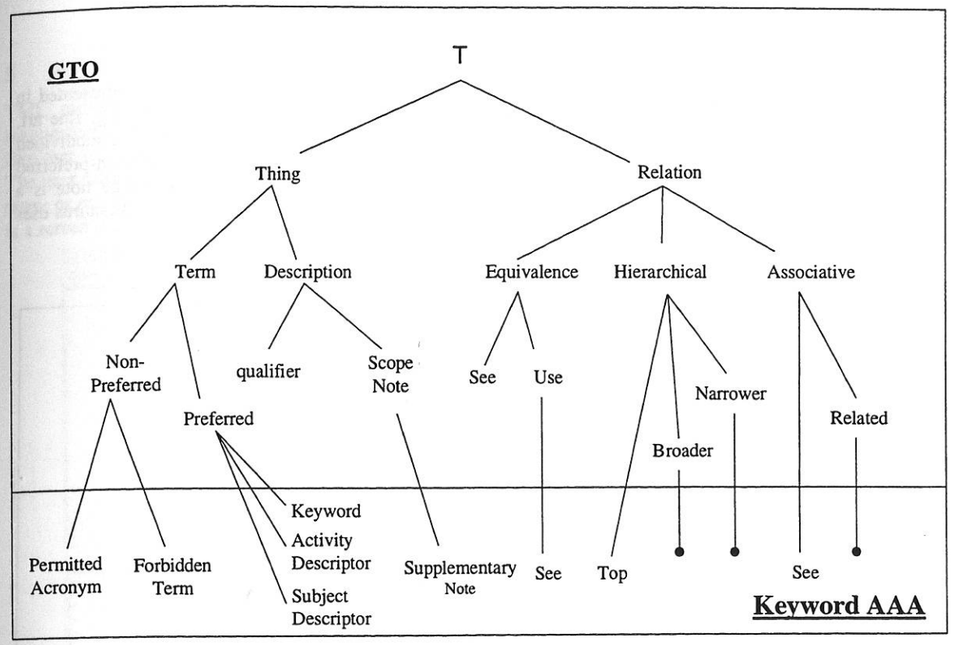


Figure 4. Keyword AAA GTO Graph

called Supplementary Note provides additional free text description. Keyword AAA employs standard hierarchical (broader & narrower) and associative (related) relations. However, the see relation in Keyword AAA is overloaded: one sense represents the equivalence use relation, and the other sense represents the associative relation. The associative relationship of the see reference is rather vague in the Keyword AAA. In some sense, it is similar to the related relation, but it may also include the near synonym and other conceptually close relationships.

Keyword AAA extends the GTO taxonomy as shown in Figure 4.
In what follows, we give examples of how Keyword AAA can be marked up in TML. The system markup element is used to record thesaural metadata.

```
<system>
<identifier version="CSIRO KeywordAAA 0.1" language="English"/>
        <authority createdBy="CMIS OMT Project"/>
        <date modifiedDate="981022"/>
</system>
```

The ontology markup element is used to define the Keyword AAA specific types extension to the general ontology (those that fall below the horizontal line in Figure 4).

```
<ontology>
        <setPrefTerm          type="KW"    name="Keyword"/>
        <setPrefTerm          type="AD"    name="Activity Descriptor"/>
        <setPrefTerm          type="SD"    name="Subject Descriptor"/>
        <setNonPrefTerm       type="PA"    name="Permitted Acronym"/>
        <setNonPrefTerm       type="FB"    name="Forbidden Term"/>
        <setScopeDesc         type="SN"    name="Supplementary Note"/>
        <setUserRel           type="TOP"   name="Top"/>
</ontology>
```

The instance markup element is used to populate the ontology structure with instances:

```
<instances>
        <prefTerm             type="KW"    value="Occupational Health &amp; Safety">
                <narrower     type="AD"    value="Policy"/>
                <nonPrefTerm  type="PA"    value="OHS"/>
                <nonPrefTerm  type="FB"    value="OHS&amp;S/>
        </prefTerm>
        <prefTerm             type="AD"    value="Policy">
                <broader      type="KW"    value="Occupational Health &amp Safety"/>
                <broader      type="KW"    value="EEQ"/>
                <broader      type="KW"    value="Establishment"/>
                <narrower     type="SD"    value="Access &amp; Equity"/>
                <narrower     type="SD"    value="Code of Ethics"/>
                <narrower     type="SD"    value="Copyright"/>
                <related      type="AD"    value="Employment Condition"/>
        </prefTerm>
</instances>
```

4.2 TML for Getty Art and Architecture Thesaurus

The Getty Art and Architecture Thesaurus [AAT] is another example of a thesaurus which can be represented in TML. AAT is a controlled vocabulary for describing and accessing cultural heritage information, e.g. fine art, architecture, decorative art, and material culture. It is structured by facets (categories), which are further subdivided into sub-facets or hierarchies. The thesaurus uses a preferred term to represent a single concept, while non-preferred terms (synonyms, spelling variants, inverted forms) are designated using a use relation and a scope note is a statement that clarifies the meaning and usage of a descriptor or guide term within the thesaurus. The thesaurus uses hierarchical relationships and equivalence relationships.

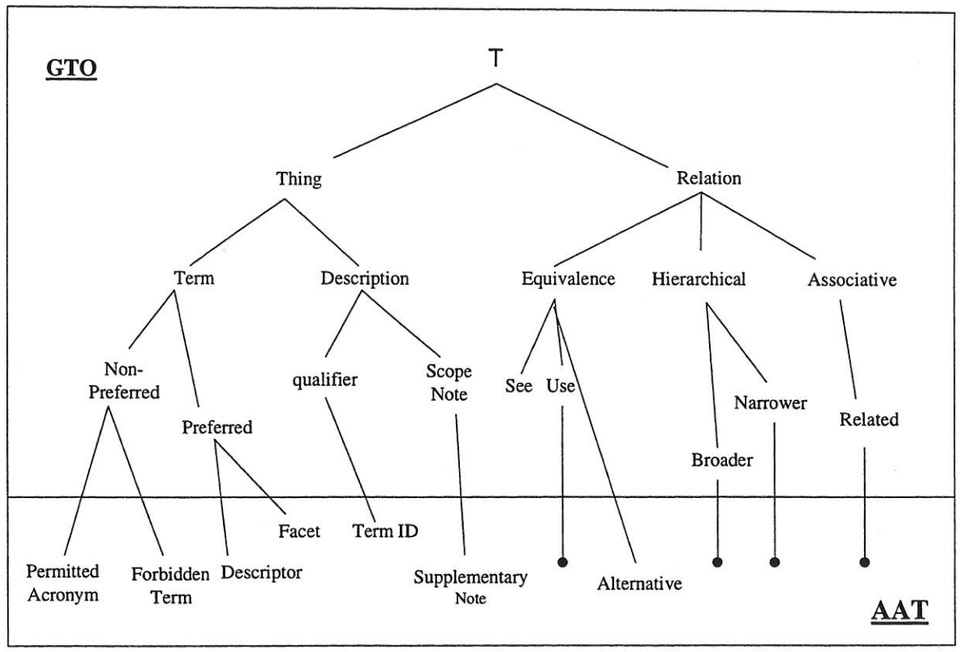


Figure 5. Getty AAT GTO Graph

AAT extends the GTO taxonomy as shown in Figure 5.
AAT can be marked up in TML using an ontology element such as:

```
<ontology>
      <setPrefTerm        type="FC"    name="Facet"/>
      <setPrefTerm        type="DS"    name="Descriptor"/>
      <setNonPrefTerm     type="PA"    name="Preferred Term"/>
      <setNonPrefTerm     type="FB"    name="Forbidden Term"/>
      <setQualDesc        type="ID"    name="Term ID"/>
      <setScopeDesc       type="SN"    name="Supplementary Note"/>
      <setUserRelation    type="ALT"   name="Alternative"/>
</ontology>
```

The ontology structure can be populated as:

```
<instances>
      <prefTerm           type="FC"    value="Associate Concepts">
            <narrower     type="DS"    value="concepts in the arts"/>
            <narrower     type="DS"    value="culture and related concepts"/>
            <narrower     type="DS"    value="environmental concepts"/>
            <qualifier    type="ID">47309</qualifier>
      </prefTerm>
      <prefTerm           type="DS"    value="concept in the arts">
            <broader      type="FC"    value="Associated Concepts"/>
            <narrower     type="DS"    value="artistic concepts"/>
            <narrower     type="DS"    value="genres in the arts"/>
            <qualifier    type="ID">56107</qualifier>
      </prefTerm>
</instances>
```


5 Application

TML provides a way to represent task-domain specific thesauri and make them available to a document management systems. In order to demonstrate this generality, we developed a *Thesaural Explorer* application. The Explorer reads a thesaurus from its TML file, presents it graphically, and supports browser style term navigation. The user selects a thesaurus to explore and then can navigate the structure along inter-term relations by clicking on terms or using various look up tables such as ordered lists by class, term alphabetic, and browsing history. Figure 6 is a screen image of the Explorer in action browsing the Keyword AAA thesaurus.

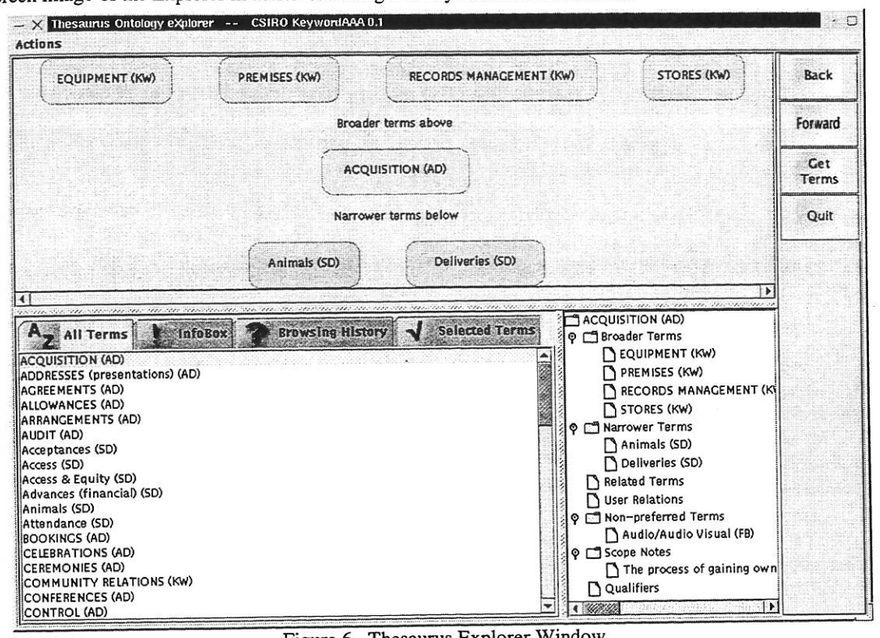


Figure 6. Thesaurus Explorer Window

The Explorer consists of three windows and a tab pane:

1. the *Terms Browser* (the window on the top) focuses on a particular term *(ACQUISITION)*, and displays its type *(AD* - in a unique hue), the hierarchical relationships between terms, and indicates if a term is selected (using intensity).
2. the *Term Viewer* (the tree window at the bottom right) shows all of the information for a particular term in a single place. This allows the user to see the full structure of a term itself.
3. the *Tab Pane* (the tabbed window at bottom left) allows the user to select from a range of information views. The default tab is "All Terms" view.

The Tab Pane shows the textual information of the Term Viewer in various presentations designed to help the user maintain browsing context:

- The *All Terms* tab has every preferred term in the thesaurus sorted alphabetically, regardless of term type. This enables examination of all preferred terms in a single list and facilities navigation to known terms.
- The *InfoBox* tab gives a textual rendition of the information graphically presented in the Term Viewer.
- The *Browsing History* tab logs all terms that have been chosen for browsing in this window so far, with the capability to move back in history and return to each term in the order they were viewed.
- The *Selected Terms* tab lists the terms that the user has chosen so far, and also has the capability of previewing them all.

6 Discussion

Our goal was to provide generic thesaural support for resource description. The GTO is not the only attempt to model thesauri. Zthes [Zthes 1999] describes an abstract model for representing and implementing thesauri. Zthes proposes the Z39.50 attribute architecture, but so far no complete implementation exists. In comparison to the TML, Zthes supports a quite restricted set of relations. It is not possible to extend Zthes in the many instances where thesauri use a wider set of relations. A limitation of the Zthes' semantics is that it does not distinguish inter term relations from term to attribute relations.

Research in ontological modelling suggests that first-order logic and other formal languages enable more precise specification of messages [Fensel *et al* 1998, Farquhar, Fikes and Rice 1996, Finin et al 1994]. However, the simplicity of the GTO does not require such powerfully expressive languages. TML comes with minimal

semantics. Semantics are modelled only at the GTO level. Beyond that, at the level of individual thesauri, semantics are user defined in syntax extensions to relation types or pushed down into the application layer. We claim this as a strength, although it could be considered as a weakness from a theoretical language stance. But it is by this simplicity that we gain our generality — by concentrating on the high-level regularities and leaving low-level peculiarities to the syntax. This gives TML a tremendous advantage over languages understood only by computer scientists. We believe that the users and maintainers of document management systems should not need to have strong technical backgrounds to do their jobs.

Our aim with TML is to represent thesauri in a practical implementable way. The TML syntax is constrained through the use of a schema, but the schema does not fully specify the language; ie., we did not attempt to include all possible thesaural semantics or to prevent all representational errors within the syntax. The verification of compliance of a thesaurus instance to the model of the GTO requires some data validation to be carried out at the application level. This is also necessarily true for the class type and relation type extensions of a particular thesaurus. This is less of a danger than it might appear, because we expect such processing validation to occur in the TML authoring tools.

The choice of XML as the TML syntax is overwhelmingly pragmatic. XML is sufficient to our requirements, an open international standard, and is emerging as a software modelling standard. It is ubiquitous and can be understood and authored without great training. It allows TML maintenance and parsing tools to leverage the power of many off the shelf authoring products.

7 Conclusions

We have demonstrated that general thesaural support is feasible by designing a generic thesaural ontology and markup language that amalgamates different thesauri structure and allows us to represent the idiosyncrasies of specific thesauri in a common language. This permits general purpose thesaural tools such as our Thesaural Explorer to be built. These tools can work with many thesauri thereby leveraging development costs, providing a common user interface, and supporting flexible thesaural maintenance and evolution. Such tools permit document management systems to better organise and access repository content.

8 Acknowledgments

The work reported in this paper has been funded in part by the Research Data Networks (RDN) Co-operative Research Centre (CRC) program, Australia.

9 References

[AAT] Getty Art and Architecture Thesaurus, URL: http://www.gii.getty.edu/aat_browser

[Bradley 1998] Bradley, N. The XML Companion, Addison-Wesley, 1998.

[Dublin Core] An International recognised core set of metadata elements. URL: http://metadata.net.de/rdf/DC/

[Farquhar, Fikes and Rice 1996] Farquhar, A., Fikes, R., and Rice, J. The Ontolingua Server: a Tool for Collaborative Ontology Construction, Knowledge Acquisition Workshop, 1996.

[Fensel *et al* 1998] Fensel, D., Erdmann, M., Studer, R. Ontobroker: The Very High Idea, *Proceedings of the 11th International Flairs Conference, 1998.*

[Finin et al 1994] Finin, T., Fritzson, R., McKay, D. and McEntire, R. KQML as an agent communication language, CIKM'94 in the proceedings of the third international conference on information and knowledge management, 1994, 456-463.

[Glushko, Tenenbaum, and Meltzer 1999] Glushko, R., Tenenbaum, J., and Meltzer, B. An XML Framework for Agent-based E-Commerce, Communication of the ACM, March 1999, Vol 42, No 3, 106, 114.

[Harold 1998] Harold, E. XML: Extensible Markup Language, IDG Books Worldwide, 1998.

[ISO 2788] International Organisation for Standardisation (ISO) 2788, Documentation Guidelines for the establishment and development of monolingual thesauri, 1986.

[Keyword AAA] *Keyword AAA: A Thesaurus of General Terms*, Archives Authority of New South Wales, Sydney, 1995.

[Lancaster 1972] Lancaster, F. , Vocabulary Control for Information Retrieval, Information Resources Press, 1972.

[LCSH] Library of Congress Subject Headings, URL at: http://www.loc.gov

[MeSH] Medical Subject Headings, URL at: http://www.nlm.nih.gov/mesh/filelist.html

[OCLC] OCLC Dewey Decimal Classification, URL at: http://www.oclc.org/oclc/fp/index.htm

[RDF] The Resource Description Framework for metadata syntax and interoperability, URL: http://www.w3.org/Metadata/RDF/

[TGN] Getty Thesaurus of Geographic Names, URL at: http://www.ahip.getty.edu/tgn_brower

[XML] Extensible Markup Language, URL: http://www.w3.org/XML/

[Zthes 1999] Z39.50 Profile for Thesaurus Navigation, URL: http://www.n-four.demon.co.uk/mirk/zthes-02.html

Building rich metadata from critical reviews for a scrutable filtering system

Sacha Groves

Basser Dept of Computer Science
University of Sydney
AUSTRALIA 2006

sacha@cs.usyd.edu.au

Judy Kay

Basser Dept of Computer Science
University of Sydney
AUSTRALIA 2006

judy@cs.usyd.edu.au

Abstract

We describe the Review Coder system for creating rich metadata for a scrutable filtering system. A scrutable system maintains explanations of the data and processes that drove the system operation. In the current paper we use Review Coder as part of a filtering systems for movies: the scrutability of the system means that a user can determine why the system recommended a particular movie or not.

The filtering process is based upon movie reviews and metadata built in association with them. These provide high quality information about the movie objects. From these, the filtering system is intended to build stereotypic models of reviewer's preferences for movies. These can drive the filtering process and the user can scrutinise both these models and the actual reviews which were used to construct them.

Keywords: Multimedia Resource Discovery, Multimedia Filtering, Scrutable Filtering, Extraction of Metadata

1. Introduction

As electronic objects become increasingly accessible, there is a growing need for tools that assist users in filtering large collections of objects so that the user can find objects of interest. The effectiveness of a filter depends significantly upon the quality of the metadata describing the objects. Accordingly, this establishes a critical role for tools which assist in the creation of high quality metadata. The importance of such tools is indicated by the vigorous efforts to create a range of tools. These tools support a range of tasks, including, for example: creation of metadata templates with tools such as Dublin Core Metadata Template (Koch, Borell, and Berggren, 1998) or the discipline specific tools like Medical Metadata Creator (, 1999); tools such as Mantis (Shafer, 1998) which manage templates and assist in production of metadata.

Filtering for large video objects such as movies is important since there is a large cost for 'browsing' such objects compared with simple text objects. This cost is both in terms of the user's time and in the bandwidth required to deliver a segment of the object suitable for browsing.

Because the effectiveness of filtering depends so heavily upon the quality of the metadata, there seems to be promise in developing collections of very rich metadata for movie objects. An indication of the interest in this area is the number of online resources about movies, such as the Internet Movies Database (Database, 1999, Guide, 1999, Finder, 1999).

Such resources can be regarded as metadata suitable for human analysis as a basis for manually selecting or filtering movies. Equally, there is considerable activity in automation of the processes, for example, development of schemas for representing movie metadata. (Hunter and Iannella, 1998, Hunter and Armstrong, 1999).

An important model for assisting in filtering involves a three stage process: define a metadata structure; allow various providers to create metadata; allow various providers to create filtering tools which operate by using the metadata. A good example of such a model is PICS (Resnick, 1998) which provides a specification for metadata intended for rating objects so parents and teachers can filter out objects which are unsuitable for their children. It

Proceedings of the Fourth Australasian Document Computing Symposium, Coffs Harbour, Australia, December 3, 1999.

enables specification of a rating service's vocabulary and scales, the format of the metadata labels and a format for filtering rules.

Note that in a PICS-style model, the filtering process will allow some objects to pass the filter because their metadata meets specified requirements. So, an explanation of the system's selection (or rejection) of an object would be in terms of three elements:

- the metadata for the object;
- the assumptions about metadata which are appropriate for a particular user;
- the actual process used by the filter to implement the intended filtering policy.

We want to build *scrutable* filtering systems, which can provide the user with access to all the elements of process so that they can determine the answer to high level questions like these. Why did the filter allow *Star Trek* through the filter? Why did it fail to allow *The King and I* through the filter? At a lower level the user should be able to find answers to questions: of these types:

A. What does the filter *believe* about me? which means: What is the filter's user model for me? and What does that model mean?

B. What does the filter *know* about *Star Trek*? which actually means: What is the metadata for this object? What does that metadata mean? What do its values mean?

C. How does the filter decide whether to filter out *Star Trek*? which means: How does the filter make use of the metadata and user model to accept or reject an object?

Our previous work on scrutable user models (Kay, 1995, 1998) provides a foundation for supporting answers to questions of type A. The current work is an extension of scrutability support. We wish to support users in understanding how the system's model of them is actually used in a system like a filter.

This paper describes a project which uses critical movie reviews as a foundation for a system which combines our goals for a source of high quality metadata for movie objects and scrutability. Essentially, our goal is that the typical critical reviews can be used to create conventional structured metadata which is suitable for use in a simple filter. At the same time, the review can serve as a basis for the explanation process associated with the metadata. So the answers to the questions of type B would take the form of: a summary of the structured metadata and the parts of the reviews which were the basis of the metadata.

Of course, for such a process to work, we need to enhance the reviews with the definition of metadata. We envisage that this might be best done by the reviewer. But it might equally be done by another person. It might even be done automatically be a natural language understanding system (although this would be a difficult problem and we are not exploring this possibility). We believe that the reviewer might be willing to invest a small amount of additional effort in creating the metadata a system might require. We also believe that there are advantages in maintaining the actual critical review in addition to the metadata. Firstly, the review is a creative work with its own value and it conveys more than the minimal information needed by a filter. Indeed, we see this project as building upon a foundation assumption that the critical review is a valuable piece of metadata for people to use. Our second reason for keeping the review as well as the structured metadata is that the review serves an invaluable role for scrutability of the system.

2. Overview of the system

Figure 1 illustrates the process involved in the first stage of the filtering system. An object such as a movie is indicated at the far left. Several reviewers write natural language free-form reviews such as those typically found in newspapers or on well known movies web sites (Ebert, 1999, Limited, 1999) as well as at our own site (Pak, 1999).

These reviews are represented in Figure 1 by the column of four reviews. The figure shows them linked from the actual movie since they are, essentially, metadata about that movie.

The next step illustrated is the definition of simple metadata tags for each review. These are shown in the middle column of the figure. Essentially, this is the production of metadata for the review: each review is the metadata for the movie as seen through the eyes of a particular reviewer. This metadata establishes each reviewer's perception of the central aspects of the movie: its positive and negative attributes, other properties and similarities with other movies. Note that one of

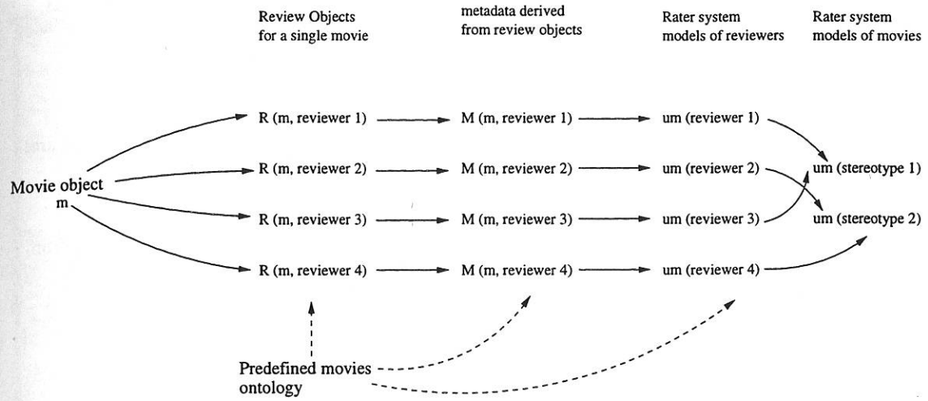


Figure 1. Overview of process to construct a stereotype for movie filtering

these reviewers might be a person involved in the production of the movie. Normally, the reviews and metadata for the reviews are produced by an independent critic.

The figure shows that the reviewer's construction of the metadata relies on a predefined movies ontology. This restricts the metadata to the set of terms which are to be used for the scrutable filter. This is important for three main reasons.

- As in the case of any filtering system, we need to ensure that the objects are coded using terms which will be usable for the subsequent filtering process and so they must be used consistently across movies and user models.
- The process is to be done by a person whose primary concern is the reviewing of a movie and we do not want to burden them with the task of deciding which terms to use.
- Finally, and critically for scrutability, we require explanations of the terms used: one of the requirements of a scrutable user model is that the user be able to access explanations of the meaning of the components of the model and these will be the terms used for the metadata. To ensure that such explanations will be available, we need to restrict the metadata to terms for which explanations have been supplied.

This stage in the review process involves a shift of perspective. We now view the metadata created by a reviewer as a source of information for modelling that reviewer. This is represented with the accretion user model representation (Kay, 1995, 1998) which models attributes of users by keeping lists of the evidence available. Although the figure shows only one movie's review and metadata feeding into the reviewer's user model, in fact, all the reviews that they have done feed into the system's model for that reviewer. So, for example, suppose reviewer 1 has written many reviews in which violence is assessed as a positive attribute of movies and has rarely assessed it as a negative aspect. Their user model would have a long list of evidence that they like violent movies and a short list indicating that they dislike it. Evaluation of the full set of evidence is performed by a resolver process: for scrutability, there must be an explanation for its operation.

The last stage illustrated in Figure 1 is the construction a small number of default models, the stereotypes (Rich, 1989) which provide a quick model for the 'average user'.

3. Metadata extraction interface

So we define two forms of metadata, which we call *bland* and *judgemental* metadata. Bland metadata is the largely objective characteristics of the movie, aspects like the directors, actors, genre and the like. These appear in typical semi-structured information about movies. Indeed, since such information is already available and classified in movie databases. We have followed their choice of metadata elements.

From our perspective, the more interesting issues lie with the judgemental metadata. This is a structured representation of the reviewer's assessment of the movie. For example, they may

have liked a particular element described in the bland metadata as in the case of a reviewer who liked the performance by Keanu Reeves in the Matrix. Such a piece of judgemental metadata is then associated with the relevant actor metadata for that movie. Actual text from the review will serve an explanation for this piece of judgemental metadata.

We allow judgemental metadata to take one of the following forms:

1. metadata about bland metadata elements (as in the example of a reviewer who liked Keanu Reeves in the Matrix);
2. metadata about the movie as a whole (and supported by that part of the review which gave that assessment);
3. metadata which compares this movie with others (for example, indicating that it is better than another, similar movie)

We want the user of our interface to make a pass over the review and review the automatically generated bland metadata, possibly altering it by deleting aspects which the reviewer does not consider important enough to serve as metadata for the movie. The reviewer may also add new metadata. Usually, however, the reviewer will use this metadata as a basis for creating the first form of judgemental metadata in the above list.

In a later pass through the metadata, the user should link each element of the metadata to the relevant text in the review.

There are two main reasons for this design. First, our concern for scrutability means that we prefer to use simpler processes for filtering if possible: these seem to be the most promising foundations for providing easily understood explanations of the system operation. Secondly, and equally importantly, we want to leave reviewers quite free to write whatever they choose and then to decide on the metadata they want to define for that review.

The Review Coder interface is, in essence, an editor for creating metadata from movie reviews. The next three figures show example Review Coder screens. The interface has two main parts. At the left is a large text area for the input of reviews. The right part of the screen is for creating the structured metadata. At the top is a list of the structured elements for the metadata. For example, Figure 2 is a start screen for developing the metadata for the film "The Matrix" where the required elements of the structured metadata are:

Identifier:matrix_the_1999
Director:Larry Wachowski
Director:Andy Wachowski

In addition, some examples of optional elements are:

Actor:Keanu Reeves (Neo)
Feature:special effects
Location:Sydney, Australia
Plot:A computer hacker suddenly learns from mysterious rebels that his whole reality is not what it appears to be at all
Similar:alice_in_wonderland
Similar:bound_1996
Similar:dark_city_1998
Similar:wizard_of_oz

Figures 2 and 3 show the interface with completed reviews. The list at the bottom right of the interface shows the metadata elements that were identified within the review with notation indicating whether the metadata element was considered

- good (+),
- bad (-),
- not as good as another film (<),
- better than another film (>)or
- as good as another film (=).

The * operator indicates that the metadata element was mentioned in the review, but that the review made no judgement about that element. For example, here is the metadata coded for a review:

+Feature:special effects
*Identifier:david_stratton
*Location:Sydney, Australia
-Plot:A computer hacker suddenly learns from mysterious rebels that his whole reality is not what it appears to be at all
*Similar:alice_in_wonderland
<Similar:bound_1996
=Similar:dark_city_1998
*Similar:wizard_of_oz

The reviewer types their review for the movie in the text box. They then select the metadata elements that they mentioned in their review from the list and code each element, one by one, using the '+','-' etc buttons.

4. Evaluation

The interface we have shown in the paper was a prototype. We have performed a small, scale formative evaluation. It aimed to assess:

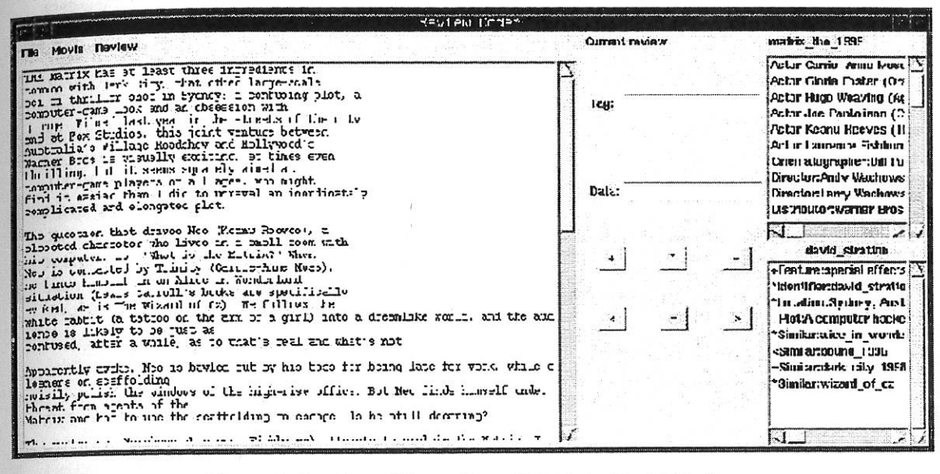


Figure 2. Interface with a review of Matrix by David Stratton

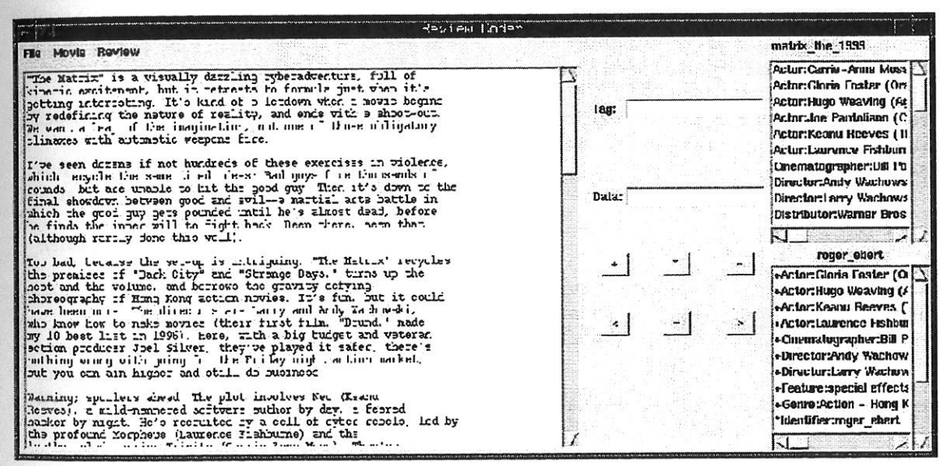


Figure 3. Interface with a review of Matrix by Roger Ebert

- the ease of use of the interface for coding a review, given a brief set of instructions;
- whether users could code a critic's review as a set of structured elements;
- the consistency in the coding between users.

The users (who we will also refer to as participants) were all asked to code a single movie review, Roger Ebert's Review of the movie "The Matrix".

The experiment took the users through the following steps.

1. Brief instructions in use of the interface.
2. The participant read the review carefully.
3. Then the participant went through the supplied metadata terms for the movie, coding the review.
4. During the process, the participants commented on what they were doing and, in keeping with the cooperative think-aloud approach (Monk, Wright, Haber, and Davenport, 1993) suggested improvements to the interface.

The participants were three Computer Science postgraduate students. All had completed a

course in user interface design and were avid movie buffs. This choice of participants means that we need to take care in interpreting this evaluation. The choice of movie buffs was desirable since we wanted the coding to be done by people who would be likely to appreciate the subtleties of the review and to take account of the broad range of elements that reviewers like Ebert mention. A side-effect of selecting movie buffs was that all had actually seen the movie before this evaluation experiment.

On the other hand, the strong computer science background of the participants makes their assessment of the interface closer to that of an expert. Since our main concern was to assess the effectiveness of the overall coding strategy, these users are a satisfactory population for a formative assessment.

Overall, the users found the interface straightforward to use. They completed the task with minimal awareness of the interface, being able to focus on the problems of deciding on the meaning of the review and how to code it. The experiment showed consistency for the '<' and '>' codings: there was no case where one user coded an aspect as '<' and another as '>'. However, different users chose to code different aspects.

5. Discussion and conclusions

The evaluation was purely formative. It indicates that users are able to attempt the task for which the interface was primarily designed: quick and simple creation of judegmental metadata.

Much remains to be done. The next phase of this research will improve the support for scrutability. The interface will be altered so that the creation of metadata will operate as follows:

- the user will be able to highlight text within the review that serves as the basis for a particular piece of metadata.
- they would then use parts of the current interface to create the metadata tag.

For example, the text "Keanu Reeves turned in a good performance" would be selected and the metadata created would set "Actor:Keanu Reeves" as "+".

Currently, the Review Coder system stores the metadata in a plain text file in the Tag:Data format. To accommodate the tagging of the evidence, we will explore use of the Resource Description Framework (W3C, 1999). Not only is RDF the current leading choice for metadata definitions, its use of XML enables the tagging of the review to be treated as a markup of the text. What is more, research into motion image metadata has suggested a Dublin Core (Core, 1999) and MPEG-7 hybrid (MPEG, 1998) using RDF as the framework to be suitable(Hunter and Iannella, 1998, Hunter and Armstrong, 1999). We are also exploring other metadata representations for objects similar to movies (CDWA, 1999). The current interface ontology is based upon analysis of metadata used by such sources as the Internet Movie Database (Database, 1999).

References

Medical Metadata Creator, http://medir.ohsu.edu/bicc-informatics/ebm/latest.htm, 1999.

CDWA, *Categories for the Description of Works of Art (CDWA),* http://hul.harvard.edu/ldi/html/metadata.html, 1999.

Dublin Core, *Dublin Core Metadata Initiative,* http://purl.org/DC, 1999.

Movie Database, *Internet Movie Database,* http://www.imdb.com, 1999.

R Ebert, *Roger Ebert on Movies,* http://www.suntimes.com/ebert/ebert.html, 1999.

Movie Finder, *Movie Finder,* http://www.moviefinder.com, 1999.

All Movie Guide, *All Movie Guide,* http://www.allmovie.com, 1999.

J Hunter and L Armstrong, *A Comparison of Schemas for Video Metadata Representation,* WWW8 http://www8.org/w8-papers/3c-hypermedia-video/comparison/comparison.html, 1999.

J Hunter and R Iannella, "The Application of Metadata Standards to Video Indexing," *Second European Conference on Research and Advanced Technology for Digital Libraries*, Crete, Greece, 1998.

J Kay, "The um toolkit for cooperative user modelling," *User Modeling and User-Adapted Interaction*, vol. 4, no. 3, pp. 149-196, Kluwer, 1995.

J Kay, *A scrutable user modelling shell for user-adapted interaction,* PhD Thesis, Basser Department of Computer Science, University of Sydney, Australia, 1998.

T Koch, M Borell, and M Berggren, *Dublin Core Metadata Template,* http://www.lub.lu.se/metadata/DC_creator.html, 1998.

News Limited, *News Limited Reviews,* http://entertainment.news.com.au/filmarc, 1999.

A Monk, P Wright, J Haber, and L Davenport, *Improving your human-computer interface: a practical approach,* Prentice Hall, 1993.

MPEG, *MPEG-7: Context and Objectives (version - 10 Atlantic City),* http://drogo.cselt.stet.it/mpeg/standards/mpeg-7/mpeg-7.htm, 1998.

J Pak, *Zoo/City Movie Reviews,* http://www.zoocity.com.au/review.html, 1999.

P Resnick, *Platform for Internet Content Selection (PICS),* http://www.w3.org/PICS/, 1998.

E Rich, "Stereotypes and user modeling," in *User models in dialog systems*, ed. by A Kobsa and W Wahlster, pp. 35-51, Springer-Verlag, Berlin, 1989.

K Shafer, *Mantis, A Flexible Cataloging Toolkit,* http://orc.rsch.oclc.org:6464/toolkit.html, 1998.

W3C, *http://www.w3.org/RDF*, 1999.

On Using Hierarchies for Document Classification

Wahyu Wibowo and Hugh E. Williams
Department of Computer Science
RMIT University
GPO Box 2476V, Melbourne 3001, Australia
{*wwibowo,hugh*}*@cs.rmit.edu.au*

Abstract

Good management of large collections, such as world-wide web databases or newswire services, is essential to ensure that they remain useful resources. Large collection management tasks include storing, querying, retrieving, routing, filtering, and classifying documents. We focus in this paper on new approaches to the last of these tasks, classification. Classification is the process of assigning one or more identifiers from a list of classes to a document. The identifier or class label is useful to organise, retrieve, or present documents. Several factors affect the effectiveness of classification schemes, including the classification method, selection of training samples, selection of features, and class label assignment methods. We identify problems in classification, propose a new evaluation framework, and show that using hierarchical information, where parent classes and subclasses of labels are used, has potential to improve classification effectiveness.

Keywords Document Management, Document Databases, Document Classification, Information Retrieval, SGML and Markup.

1 Introduction

Large online collections continue to grow in volume and to place demands both on space resources and on techniques to manage and query databases. For querying of world-wide web databases, Boolean and ranked querying remain the most popular techniques for finding relevant documents to our information needs. However, alternative techniques that offer different ways of exploring information, such as the Yahoo! search engine [1] hierarchical approach or browsing with phrases [15], are also proving useful methods to satisfy our search needs.

Text classification, where documents are assigned one or more labels from a predefined set, is one possible technique that can be used to improve the organisation and management of data. In the case of Yahoo! and in many specialist domains—such as keyword assignment in the Medline bibliographic database [5] or in the GenBank genomic database [2]—such classification is a manual process, where trained experts assign new documents to classes and organise classes into a hierarchy where each node can potentially have multiple parents and children. Building such knowledge bases, while successful and popular for specialist applications, requires significant effort in development.

Automatic class label assignment, without the need for human intervention in the classification process, is desirable for speed, scalability, and cost. However, automatic classification is an uncertain and difficult process. Automatic techniques to accurately classify documents must identify the characteristics of documents that belong to a certain class, suppress noise that may affect the assignment judgement, and be able to determine the class or classes of a document based on the available data.

In automatic classification *training* with sample documents is used to develop a *classifier*. The sample documents in this approach are manually annotated with class labels and the classifier trained to identify the characteristics of each document class. After the training process is complete, unclassified documents are processed and compared to the statistics and characteristics of the training documents to identify features and assign class labels. Automatic classification, in contrast to manual classification, has not been used to classify documents into a hierarchy, but has only been used to classify documents into one or more classes from a set of classes.

We experiment in this paper with classification of documents based on a training set derived from the Reuters newswire service and consider whether the effectiveness of this classification approach can be improved by considering the relationships between different class labels. Specifically, we extend previous experiments and propose a new measurement framework to show that by arranging the class labels in a *hierarchy*, where a document can

[1] `http://www.yahoo.com`

Proceedings of the Fourth Australasian Document Computing Symposium, Coffs Harbour, Australia, December 3, 1999.

be classified into a tree structure of parents and children, that the accuracy of classification may be able to be improved by considering the relationship between parent and child nodes in the class label tree structure. Our conclusion is that classification of documents into broader parent classes is more accurate than classification into child classes. We expect that hierarchical classification, where parent classification information is used to aid child classification, will improve the accuracy of automatic document classifiers.

2 Document Classification

Development of a trained classifier for automatic classification of documents first requires a training technique based on the pre-classified documents. In deciding how to train the classifier, two questions must be addressed: first, what features of the documents will be derived and used to train the classifier to recognise documents; and, second, how will features be represented and used in the classifier? We discuss these two questions in this section and begin by considering techniques for feature extraction from training documents.

In document retrieval systems, one common way to represent a document is by viewing the document as a collection of features in the form of *words* or *terms*, that is, as unit strings of characters in a document that are separated by white space characters. Using terms, a document can then be represented as a feature space using a vector model $(i_1, i_2, \ldots, i_n)$ where each element i_j is either 0 or 1 depending upon the existence of a term in a document, the occurrence frequency of a term in the document, or *weights* that reflects the importance of an index term. Other approaches to representing documents include using structure or mark-up information to represent a document, but we do not discuss these here.

Many schemes have been proposed for weighting terms within a document, that is, representing the significance of a term in a document in such a feature space model. Most schemes are based on variations of the TF.IDF [11] measure in which the importance of a term in a document is the product of the frequency of the term in the document and the inverse of the number of documents that contain this term. In this way TF.IDF is often calculated as

$$\text{TF.IDF}(i,j) = \text{tf}_{i,j} \times \log(\frac{N}{\text{df}_i})$$

where $\text{tf}_{i,j}$ is the term frequency of term i in document j, df_i is the number of documents that contain term i, and N is the number of documents in the document set.

This approach of representing a document can be generalised to a scheme for representing a class of documents that have the same features. By selecting all training documents that are pre-assigned to a class, it is possible to represent the terms in a class of documents as a vector $c_i = (td_{i,1}, td_{i,2}, \ldots, td_{i,n})$ where c_i is the represented document, $td_{i,j}$ is the j-th term descriptor in document i, and n is the number of term descriptors. This vector then is a descriptor of features of a specific class and, as we discuss later, comparison of unclassified documents to such a vector can be used for classification. These approaches to *linear classification* have been shown elsewhere to be an effective method of developing class features from training sets [1, 7, 9]; we describe details of one of these approaches, that of Lam and Ho [7], in the next section.

In representing documents as a vector, some words or terms may be *stopped* in the document identification process [16]. These words are usually common words used by almost all type of documents such as articles (for example "a" or "the"), prepositions (for example "to", "for", or "at"), or very common words (for example "while", "if", "else", "before", or "after"). In addition to consuming processing time in classification, when such terms are removed documents may be more separable for class identification. The terms are eliminated using a static list of stop words, or using a feature selection algorithm such as document-frequency thresholding, information gain criteria, term-strength criterion, mutual information, and χ^2-Tests [17]. The results presented in this paper use simple document-frequency thresholding to remove common terms.

2.1 Rocchio Classifier

The approach to representing a class of documents as a vector, as described in the last section, is both simple and practical. However, this scheme only considers positive information, where the presence of a term in a document class adds that term to the vector. Another approach is to consider negative information, where the presence of a term in a different class reduces the weight or importance of a term. In this way, all terms in the collection are represented in a class vector, with terms present in the class typically having positive weights and those not present in the class typically having negative weights.

Several possible candidate schemes exist for feature representation that includes negative and positive weights [7, 9, 10, 14]. We use the Rocchio weight learning technique [10] which, while having been shown to be marginally less effective for classification than other approaches [7], is simple to implement and practical for our experiments in studying hierarchical classification techniques.

The Rocchio approach is based on a simple similarity measure, where the similarity of two vectors is computed; in this case, the two vectors are for a new training document and an existing vector that represents a class. Such *linear similarity* is computed with a dot product of the two vectors w, the weight vector for a class, and x, the vector of the training document, so that:

$$f(x) = w \cdot x = \sum_{j=1}^{d} w_j x_j$$

where d is the number of term descriptors in vectors w and x.

The Rocchio measure uses the linear similarity measure as follows: for each new training document, the class representative vector for each class is modified by adding the weight of the linear similarity of the positive training terms and subtracting the weight of the linear similarity of the negative training terms. For a class representative vector w and a new training document x, the Rocchio measure is:

$$new\ w_j = \alpha w_j + \beta \frac{\sum_{i \in C} x_{i,j}}{|C|} - \gamma \frac{\sum_{i \notin C} x_{i,j}}{n - |C|}$$

where n is the total number of training examples, C is the set of positive training examples, and α, β, and γ are constants. In our experiments, we use constants of $\alpha = 1$, $\beta = 16$, and $\gamma = 4$, the same as those reported by Lewis et al. [9].

Given the Rocchio measure, it is then a simple batch process to derive a class feature vector for a given set of classes and training documents. For each training document, the weight of the vector of each class is modified so that when a document is a class member the weight of class terms are increased and when a document is not a class member the weight of non-class terms is decreased. The result is a vector for each class w of length t, where t is the count of distinct, unstopped terms in the collection; in this way, all class vectors are of length t and contain all unstopped terms.

3 Training Documents

Given a Rocchio classifier based on the TF.IDF weighting scheme, as described in the last section, we need to consider suitable test collections, how classes are trained, and how unclassified documents will be compared to the class feature vectors by the classifier. Most importantly, we need to consider how to measure when a test document is assigned or not assigned to a class.

3.1 Reuters Test Collection

Several test collections have been used as training and test sets for linear text classifiers, including the AAP Newswire [3, 8], the MEDLINE Database [17], the Ohsumed database [7], and the Reuters Collection [3, 6, 7, 18]. In our experiments, we use the Reuters-21578 text categorisation[2] test collection of Reuters newswires from 1987 to 1991[3]. This collection contains 21,578 SGML articles stored in 22 data files. Each article in the collection is headed with a tag of the form:

<REUTERS TOPICS=?? LEWISSPLIT=??
CGISPLIT=?? OLDID=?? NEWID=??>

Some documents in the Reuters-21578 collection have no class assignment information in the article header, while others contain irrelevant information. Such documents will not be useful for classification experiments and, because of this, several different splits or divisions of this collection have been proposed for research experiments. Each division is based on the value of the starting tags and we follow the approach of deriving a split between training and test documents proposed by Apte et al. [1], the so-called "Mod-Apte" division.

In "Mod-Apte", the Reuters-21578 collection is divided into three sets:

1. "Training Set": 9,603 documents with the following tags: LEWISSPLIT="TRAIN"; TOPICS="YES"
2. "Test Set": 3,299 documents with the following tags: LEWISSPLIT="TEST"; TOPICS="YES"
3. "Unused": 8,676 documents with the following tags: LEWISSPLIT="NOT-USED"; and either:
   - TOPICS="YES" or
   - TOPICS="NO" or
   - TOPICS="BYPASS"

The documents in the training and test sets in the "Mod-Apte" divisions contain class assignments, where the class assignments are hierarchical. At the *parent* level, there are six classes: "COMPANIES" (which has no positive training examples), "TOPICS" (7,775 positive examples), "PLACES" (8,959 positive examples), "ORGS" (456 positive examples), "PEOPLE" (433 examples), and "EXCHANGES" (73 positive samples). At the *child* level, where each child is associated with one parent class, there are 368 classes, including topics as diverse as "ALGERIA", "AMEX", and "ALUM(inium)". Several child classes have more than 1,000 positive training examples, while 80 classes have only one positive training example. Note that documents can be

[2]The term "categorisation" is sometimes used to describe the process of assigning predefined class labels to documents. We use the term "classification" throughout to describe this process.

[3]The Reuters-21578 collection is available at `http://www.research.att.com/~lewis`

classified into multiple parent and child classes and can therefore be positive examples for multiple classes, as well as always being negative examples for the remaining classes.

3.2 Training the Classifier

In previous experiments [7], linear classification has focused on evaluation with the children of one parent class, for example "TOPICS", using only child classes with more than one positive training example. Our focus in this paper is expanding this approach to investigate the classification of documents into both parent and child classes. In training our classifier, we use all 9,603 documents in the "Mod-Apte" training set and generate class feature vectors for each of the parent classes and child classes, giving a total of 374 vectors each with a length or number of weighted terms of just over 20,000. In our experiments, which we describe in the next section, we compare each of the 3,299 test documents to each of the 374 feature vectors.

3.3 Classifying Documents

Given a trained classifier and a set of test documents, the question remains as to how we decide how does a document fit in a hierarchy of classes and can hierarchy information be used to better classify documents? A simple approach, and the one we employ, is to quantify classification by calculating a vector for each test document using TF.IDF and then calculate the dot product of the test document vector and each of the parent and child feature class vectors. The results, which are similarity scores between the test document and the parent class feature vectors, and similarity scores of the test document to the child class vectors, can then be ranked in order of decreasing similarity. In our case, in contrast to previous approaches, further assessment of the accuracy of these rankings is a particularly difficult problem, as each document can be specified as belonging to multiple parent and child classes.

Several different approaches have been proposed for quantifying the accuracy of rankings of linear classification schemes [18]. One of the more popular approaches is to calculate a "breakeven recall-precision point", where: the number of true positive classifications, true negatives, false positives, and false negatives are summed; standard recall and precision calculated; and, values interpolated to give a breakeven point (a point where recall and precision are the same) [7]. This "breakeven point" approach has been criticised as being artificial [12], since the point calculated is interpolated and represents a point not achievable by the system.

We propose a new technique for evaluating the performance of classification into the parent-child hierarchy of the Reuters-21578 collection by using the number of expected answers as a cut-off point for assessment, a similar approach to that of R-precision [4] or missed-at-equivalence [13]. In this new approach, the number of expected answers is the number of parent or child classes that a test document is assigned to. We use this number as a cut-off point for measuring the number of correct and incorrect assignments.

To illustrate this approach, consider the ranking of a test document against the feature parent classes, where the test document has been manually assigned by a human judge to two parents "PLACES" and "TOPICS". After the comparison with the Rocchio classifier, the similarity to the feature class vectors returned is ranked in the order: "PLACES", "PEOPLE", and then "TOPICS". As the number of correct parents is two, we use a cut-off of two to assess the parent answers; in this case, the first two answers contain one correct parent response "PLACES" and one incorrect parent response "PEOPLE". Similarly, for the children the manual assignment is of eight classes: "trade" (a child of "TOPICS"), "malaysia" (PLACES), "south-korea" (PLACES), "australia" (PLACES), "hong-kong" (PLACES), "usa" (PLACES), and "japan" (PLACES).

We combine our cut-off assessment of parents with a cut-off assessment of child rankings. In the left-most column of Table 1, a ranking is shown of the similarity of the test document to the child feature classes using our classifier; the parent class of each of the ranked results is shown in the middle-column. As with the parent assignments, we cut-off the list at the number of correct child assignments.

The right-most column of Table 1 shows the results of our assessment using this cut-off approach. For each answer, we assess whether the parent of the child was correctly identified above the cut-off—if the parent was identified we say that "P=T(rue)" and otherwise "P=F(alse)". We also assess whether the child answer was correctly identified above the cut-off by indicating "C=T(rue)" or "C=F(alse)". After assigning a value to "P" and "C" for each child above the cut-off, we sum the results of each of the four combinations of "P" and "C" values.

In our example in Table 1, there are five "P=T C=T", zero "P=T C=F", one "P=F C=T", and two "P=F C=F". An assignment of "P=T C=T" is a correct assignment: we have successfully identified a child class above the cut-off and also correctly identified the parent class of that child above the cut-off. The opposite case is "P=F C=F", where the child is incorrectly identified and the parent is also not correctly identified (either a false positive or false negative). The two remaining cases are partially correct identification. The first case is a correct parent, where we identify an incorrect child, that is, the identified child is a sibling of a correct child. The second case is a correct child, where we have not correctly identified its parent.

Ranked Child Results	Parent	Assessment
trade	TOPICS	P=F, C=T
japan	PLACES	P=T, C=T
hong-kong	PLACES	P=T, C=T
south-korea	PLACES	P=T, C=T
taiwan	PLACES	P=T, C=T
gatt	ORGS	P=F, C=F
usa	PLACES	P=T, C=T
yeutter	PEOPLE	P=F, C=F

Table 1: *Sample ranking of child feature classes for a test document. Eight child assignments were made in a manual assessment and we show the top eight ranked reponses from the classification process. The parent of each ranked child is shown in the middle column. The third column shows an assessment: "P=T(rue)" when we have identified the parent of this child in our parent ranking and this agrees with the manual assignment; "C=T(rue)" when we have correctly identified a child matching the manual assessment. The ranking of parents has identified one of two correct parent classes, "PLACES", but not identified the class "TOPICS".*

	Correct Child	Incorrect Child
Correct Parent	5,973	1,484
Incorrect Parent	126	486

Table 2: *Summed classifications of test documents into a hierarchy derived from the Reuters-21578 test collection using 3,299 test documents on a trained set of 6 parent classes and 368 child classes. "Correct" means that the class was ranked above the cut-off and incorrect indicates the class was not ranked above the cutoff.*

4 Hierarchical Classification

Using our approach of analysing the results of hierarchical classification, it is possible to quantify and analyse the likelihood of hierarchies improving the accuracy of classification. The results of our analysis of child-parent classification is shown in Table 2.

Our results shown in Table 2 show the process of testing 3,299 documents against the hierarchy of 6 parent classes and 368 child classes. In these tests, 8,069 manual class assignments were made by human assessors, an average of 2.4 parent and child classes per document. Of these manual class assignments, 74% were classified correctly by our Rocchio classifier both into the correct parent and correct child class, that is, we had identified a child and also correctly identified its parent.

The cases of most interest in our hierarchy are those where either or both the parent and child classification fails. Around 6% of classifications were complete failures, where a child of an unrelated parent was identified. Around 18% were sibling identifications, where an incorrect child is identified of a correct parent, and the remaining around 1% were cases where we identified a correct child but had not identified its parent.

Our results show that when classification fails, it most often fails in child classification. Indeed, in our experiments identification of a sibling is more than 11 times more likely than incorrect parent identification. This is not surprising, since parent classes represent broader topic areas than child classes and we therefore expect that parent vectors are more likely to be separable and distinct than are child vectors.

The skewing of classification failure suggests that parent classes can be used to improve the classification into child classes. If a parent class is much more likely to be identified correctly than a child class, then the parent information can be used to adjust the rankings of children to prefer children that are members of the parent class. Such an approach will lower the ranking of incorrect children, improving the rankings of both correct children and siblings, and reducing both the total failure and the correct parent/incorrect child figures.

Note that we have also observed, as have others [7], that classification failure is more likely when few training documents are available. As an example, many incorrect parent assignments are made to the class "COMPANIES", which has no positive training documents, and many incorrect child assignments are made to classes with one or a only a few positive training examples.

5 Conclusion

Classification of documents into classes is one technique to improve the management of large collections to ensure they remain useful resources. Indeed, document classification by assignment of identifiers from a predefined list is a valuable tool to organise, retrieve, or present documents. Such classification is generally performed with simple linear classifiers, where documents are classified into one or more classes by a pre-trained classifier.

We have presented in this paper a new framework for evaluating classification, which is based on analysis of classification of documents into hierarchies. We have identified problems in classification and shown that using hierarchical information, where parent classes and subclasses of labels are used, has potential to improve classification effec-

tiveness. Indeed, while the assignment process is somewhat affected by the training sets available, the simple ranking algorithm used, and the use of simple Rocchio, our new framework focuses on the relative success of classification into parent and child classes.

Our results indicate that parent classification is more likely to be successful than child classification because of parent classes covering broader, more distinct topic areas. By using parent classifications, which are more likely to be correct, it is likely that hierarchies may be an aid to improving classification in the more fine-grain child classes and removing false-positives. We are currently developing new techniques based on this analysis of classification into hierarchies and believe that hierarchical classification is a valuable tool in improving the accuracy of classification techniques.

Acknowledgments

We thank Ross Wilkinson for valuable discussions and the referees for their feedback. This work was supported by the Multimedia Database Systems group at RMIT University.

References

[1] C. Apte, F. Damerau and S. Weiss. Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, Volume 12, Number 3, pages 233–251, 1994.

[2] D.A. Benson, M.S. Boguski, D.J. Lipman, J. Ostell and B.F. Ouellette. GenBank. *Nucleic Acids Research*, Volume 26, Number 1, pages 1–7, 1998.

[3] W.W. Cohen and Y. Singer. Context-sensitive learning methods for text categorization. pages 298–306, New York, August 18–22 1996. ACM Press.

[4] D. Harman. Overview of the second text retrieval conference (TREC-2). *Information Processing & Management*, Volume 31, Number 3, pages 271–289, 1995.

[5] W.R. Hersh and R.B. Haynes. Evaluation of SAPHIRE: an automated approach to indexing and retrieving medical literature. In *Proceedings of the 15th Annual Symposium on Computer Applications in Medical Care*, pages 808–812, 1991.

[6] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *European Conference on Machine Learning (ECML)*, 1998.

[7] W. Lam and C.Y. Ho. Using a generalized instance set for automatic text categorization. In R. Wilkinson, B. Croft, K. van Rijsbergen, A. Moffat and J. Zobel (editors), *Proc. ACM-SIGIR International Conference on Research and Development in Information Retrieval*, pages 81–89, Melbourne, Australia, July 1998.

[8] D.D. Lewis and W.A. Gale. A sequential algorithm for training text classifiers. In W. Bruce Croft and C.J. van Rijsbergen (editors), *SIGIR 94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 13–12, London, 1994. Springer-Verlag.

[9] D.D. Lewis, R.E. Schapire, J.P. Callan and R. Papka. Training algorithms for linear text classifiers. In Hans-Peter Frei, Donna Harman, Peter Schäuble and Ross Wilkinson (editors), *Proc. ACM-SIGIR International Conference on Research and Development in Information Retrieval*, pages 298–306, New York, August 18–22 1996. ACM Press.

[10] J.J. Rocchio. Relevance feedback in information retrieval. In *The Smart Retrieval System — Experiments in Automatic Document Processing*, pages 313–323. Prentice-Hall, Englewood, Cliffs, New Jersey, 1971.

[11] G. Salton (editor). *The SMART Retrieval System—Experiments in Automatic Document Processing.* Prentice-Hall, New Jersey, 1971.

[12] R.D. Schapire, Y. Singer and A. Singhal. Boosting and rocchio applied to text filtering. In R. Wilkinson, B. Croft, K. van Rijsbergen, A. Moffat and J. Zobel (editors), *Proc. ACM-SIGIR International Conference on Research and Development in Information Retrieval*, pages 215–223, Melbourne, Australia, July 1998.

[13] E.G. Shpaer, M. Robinson, D. Yee, J.D. Candlin, R. Mines and T. Hunkapiller. Sensitivity and selectivity in protein similarity searches: A comparison of Smith-Waterman in hardware to BLAST and FASTA. *Genomics*, Volume 38, pages 179–191, 1996.

[14] B. Widrow and S.D. Stearns. *Adaptive Signal Processing.* Prentice-Hall, Englewood Cliffs, NJ, 1985.

[15] H.E. Williams, J. Zobel and P. Anderson. What's next? Index structures for efficient phrase querying. In John Roddick (editor), *Proc. Australasian Database Conference*, pages 141–152, Auckland, New Zealand, January 1999.

[16] I.H. Witten, A. Moffat and T.C. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images.* Van Nostrand Reinhold, New York, 1994.

[17] Y. Yang. Expert network: Effective and efficient learning from human decisions in text categorization and retrieval. In W. Bruce Croft and C.J. van Rijsbergen (editors), *SIGIR 94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 13–22, London, 1994. Springer-Verlag.

[18] Y. Yang. An evaluation of statistical approaches to text categorization. Technical report, CMU-CS-97-127, Carnegie Mellon University, 1997.

Documenting Business: The Australian Recordkeeping Metadata Schema

Glenda Acland
glenda.acland@uq.net.au

Barbara Reed
Barbara.Reed@recordkeeping.com.au

Ass. Prof. Sue McKemmish
Sue.McKemmish@sims.monash.edu.au

Records Continuum Research Group
School of Information Management
& Systems
Monash University
26 Sir John Monash Drive
Caulfield East, Victoria 3145

Abstract

In July 1999, the Australian Recordkeeping Metadata Schema (RKMS) was approved by its academic and industry steering group[1]. This metadata set now joins other community specific sets in being available for use and implementation into workplace applications. The RKMS has inherited elements from and built on many other metadata standards associated with information management. It has also contributed to the development of subsequent sector specific recordkeeping metadata sets. The importance of the RKMS as a framework for 'mapping' or reading other sets and also as a standardised set of metadata available for adoption in diverse implementation environments is now emerging. This paper explores the context of the SPIRT[2] Recordkeeping Metadata Project, and the conceptual models developed by the SPIRT Research Team as a framework for standardising and defining Recordkeeping Metadata.[3] It then introduces the elements of the SPIRT Recordkeeping Metadata Schema and explores its functionality before discussing implementation issues with reference to document management and workflow technologies.

Introduction

Metadata has existed in record systems throughout time. Metadata, which can be generically defined as 'structured data about data', is simply a new term for the type of information that has always been captured in records and archives systems. But it is only now that the recordkeeping community has begun the process of the codification of recordkeeping metadata so it can be fully understood and employed both within and beyond our own profession.

Within the context of various metadata related initiatives in Australia and elsewhere, the SPIRT Recordkeeping Metadata project was envisaged to build a framework in which other sector specific metadata standards could be developed for targeted application.[4] The major deliverable of the eighteen month Research Project, ***Recordkeeping Metadata Standards for Managing and Accessing Information Resources in Networked Environments Over Time for Government, Commerce, Social and Cultural Purposes*** is the Australian Recordkeeping Metadata Schema (RKMS), a high level extensible framework for specifying, standardising and mapping recordkeeping metadata[5]. Work is now proceeding on related research deliverables, including metamodelling of the schema in RDF[6] and ORM[7], the development of a User Guide to the set, and a prototype recordkeeping system that deploys the RKMS.[8]

The Context of the SPIRT Recordkeeping Metadata Initiative

In response to the policy directions announced in late 1997 as part of the Australian Government's Investing for Growth strategy[9], a range of initiatives has been taken to support and encourage individuals and organisations to transact business electronically. They include initiatives relating to the establishment and accessibility of online government services and call centres. Information resource management initiatives are addressing challenges relating to dealing interoperably at the global level with facilitating resource description and discovery.

The thrust of government online initiatives is towards fully enabled online transactions as a significant component of service delivery. The *Electronic Transactions Bill 1999*[10] is a model law which potentially provides the regulatory framework for the use of electronic communications in transactions (defined broadly to encompass all of the activities of government agencies in their roles as service providers). In the environment envisaged by the Bill, services will need to be documented and

Proceedings of the 4th Australasian Document Computing Symposium,
Coffs Harbour, Australia,
December 3, 1999.

instances of service delivery will need to be recorded. They will need to be clearly linked to what the agencies responsible for the services are mandated to do (what functions, activities, and transactions they are responsible for carrying out). The concerns of the national online service communities, which are closely linked to issues of client confidence, include the need for harmony, interoperability, improved access in global networked environments to services and service delivery points; and reliable, authoritative, trustworthy information about services.

To support an enterprise's business functions and activities in cyberspace, and to ensure the persistence and continuing accessibility of records of those activities that are of long term value to society, innovative, reliable and robust mechanisms are required to enable the continuing reliability and accessibility of essential evidence of business activities. Electronic recordkeeping and archival systems provide such mechanisms. They are fundamentally concerned with identifying, describing and classifying the functions, activities and transactions that records document. This can be contrasted with the fundamental concern with subject classification in library and information systems. Records document actions, not subjects – they record what an organisation does – its business transactions, the business activities of which they are a part, the business functions the activities carry out, and the broader societal purposes they fulfil. Linking records to the functions, activities and transactions they document and the agents involved is fundamental to establishing:

- who has the competence or authority to undertake the business activity or deliver the service,
- who is responsible for the business activity or service delivery,
- what instrument authorises action,
- how to access accountable information (records) about business transactions or instances of service delivery.

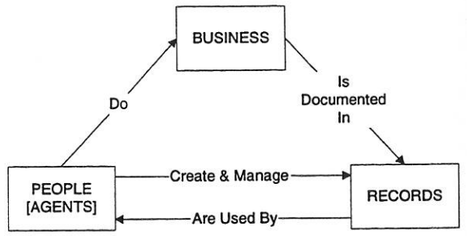
The recordkeeping community in Australia has been vitally concerned with the quality of public and corporate recordkeeping in electronic environments, recordkeeping-related issues concerning the reliability, accessibility and accountability of online activities and services, and the persistence and accessibility of records of continuing value to society. Major problems in electronic recordkeeping have been linked to the lack of controls, frameworks and standards in this rapidly evolving area. The response has been a proactive, innovative approach to the research and development role, epitomised in the involvement of the industry partners in the SPIRT 1998-99 Research Project.

The broader social context of the project relates to enabling society, government, commerce and individuals to continually access the information they need to conduct their business, protect their rights and entitlements, and securely trace the trail of responsibility and action. Maintaining authentic, reliable and useable evidence of transactions has significant social and cultural implications as records are a bastion of democratic and cultural accountability. They enable democratic rights of review and examination, and the transmission of our cultural heritage.

Framework for Standardising and Defining Recordkeeping Metadata

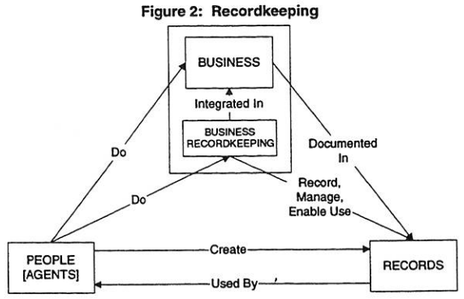
The Research Team has developed three high level models, Figures 1, 2 and 3, to provide the framework for standardising and defining recordkeeping metadata.[11]

Figure 1: The Business



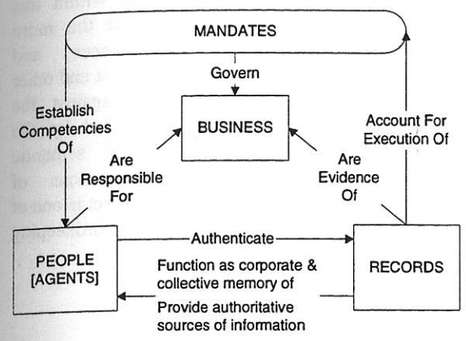
When people do business with each other, they create and manage records. The records created in the course of doing business capture the business done in documentary form. Business is here defined in the very broadest sense to encompass social and organisational activity of all kinds. A simple way of defining a record therefore is as a document that has taken part in a business process, and thereafter provides evidence of the transaction of that business. In distributed systems environments, records form a significant subset of an enterprise's digital information and knowledge resources.

Figure 2: Recordkeeping



Optimally recordkeeping forms an integral part of any business activity whatever technology is used.

Figure 3: The Business Context



People do business in social and organisational contexts that are governed by external mandates (e.g. social mores, laws, regulations, standards and best practice codes) and internal mandates (e.g. policies, administrative instructions, delegations, authorities). Mandates establish who is responsible for what, and govern social and organisational activity, including the creation of full and accurate records. Authentic records of social and organisational activity provide evidence of that activity and function as corporate and collective memory. They also provide authoritative sources of value added information. And they account for the execution of the mandate – internally and externally, currently and over time.

Recordkeeping Metadata

With reference to these high level conceptual models the RKMS is presented diagramatically (Figure 4) as essentially concerned with three classes of entities – **Business** entities, **People/Agent** entities and **Records** entities, *as well as with* the external and internal **mandates** which are associated with Business, People and Records entities and govern the relationships between them. Furthermore, **Business-Recordkeeping** entities form a sub-class of the **Business** entity class.

The RKMS envisions management of records, agents and business at different layers of aggregation. A taxonomy of layers has been defined. (See Figure 4)

All these entities and their complex inter-relationships require unique identifiers and standardised descriptive metadata. The RKMS enables relationships to be set up between the layers of agent, business and record in addition to relationships within the layers. Any single record may have relationships which extend through layers of aggregation in ways which establish a rich envelope of contextual metadata.

This complexity in relationships and their fundamental importance in defining the records context has been pushing beyond the requirements of other information resource metadata sets. While the conceptual understandings of relationships is well developed, issues to do with the taxonomy of relationships, the precision of the depiction of relationships and the metadata expression of such relationships is a further fruitful area for future research.

The Recordkeeping Metadata Elements Schema – Version 3.02

A highly structured set of elements and qualifiers has been defined (note that only the elements are represented in Figure 5). The view of the Schema provided in Figure 5 presents the elements in four sub-sets. This view is derived from the conceptualisation of records in their business context as depicted in Figures 1-3 above. The RKMS inherits part of the Australian Government Locator Service set and extends it to address the sector specific needs of recordkeeping.[12]

The elements and qualifiers defined in the Recordkeeping Metadata Schema identify and describe significant features of the business contexts in which records are created, managed and used. They identify and describe the people or agents involved, and the records themselves. They also link business contexts to the people or agents doing the business and the records that document it, and they reference the mandates that authorise and control business activity. They enable description and management of recordkeeping business functions, activities and transactions that are concerned with recording, managing and enabling the use of records, e.g. transactions and activities relating to the recordkeeping functions of appraisal, control, preservation, retrieval, access and use of records. They also provide for the tracking and documenting of the recordkeeping business itself in the unique metadata elements associated here with the **Records** entity.

Qualifiers in the RKMS

The RKMS qualifiers allow for a more detailed recordkeeping description, providing the facility to refine the semantics of the RKMS and to add precision to the values of the metadata elements. The RKMS has adopted the DC/AGLS application of three types of qualifiers,[13] element qualifiers, value components and value qualifiers. The metadata

community is only beginning to explore the complexity of the schemes which govern and control metadata values.[14]

Scalability

As mentioned above, a significant feature of this high level set of metadata is that it is scalable, i.e. when it is implemented it can apply to records at any level of aggregation, to business and recordkeeping business activities ranging from an individual transaction to the societal purpose it ultimately serves, and to agents acting at any level in organisational and social hierarchies. An Entity "switch" has therefore been included in the set. In any particular instance the Entity Switch indicates whether a **Business**, **Recordkeeping Business**, **Agent** or **Records** entity is being described. Within each entity, the *CATEGORY TYPE* element then functions as a handshake, introducing the specific type of entity being identified and described:

Extensibility: Inheritance of Metadata

The RKMS envisages use of metadata elements, element qualifiers and value components from other metadata sets. Within individual elements, it is possible to extend the RKMS specification by referencing other schemata, e.g. the Pittsburgh Business Acceptable Communications[15] Structure layer metadata elements and qualifiers could be used to extend the **Records**: PRESERVATION and **Records**: RETRIEVAL elements. Indeed the RKMS could inherit a full range of metadata elements, qualifiers, value components and prescribed schemes from another metadata schema for one of its entities.

The RKMS also envisages inheritance of the data values from another schema. Particularly when specifying metadata associated with agents and business, it does not seek to create separate recordkeeping views of these entities. Rather it enables reference to metadata sets defined in other circumstances. The RKMS also provides for the definition and an external validation of authority for such inherited sets.

A Framework for Mapping Metadata

One significant component of the research activity undertaken during the project has been an in-depth analysis of existing records and archives metadata schemata and standards. This was accompanied by the conceptual mapping of their elements in various combinations, followed, as the project advanced, by mapping the various iterations of the Schema against these related sets. The mapping processes which informed the development of the RKMS metadata set itself, point to one of its major uses - as a framework in which other sets, targeted for application in specific sectors, can be developed and mapped. For example, the National Archives of Australia's *Recordkeeping Metadata Standard for Commonwealth Agencies*, released in June 1999[16] was developed within this framework and can be mapped against the more comprehensive RKMS. Equivalences and correspondence can thus be made between it and other metadata sets, each one being read against the standardised metadata framework provided by the SPIRT Schema. The capacity for semantic interoperability of specific implementations of metadata when mapped against a standard set is one of the resulting benefits for the recordkeeping community, nationally and internationally.

Documenting Business

The RKMS as presented in this paper is modelled conceptually. As yet no implementation models have been attempted, although the metamodelling in RDF will enable the expression of the metadata in XML and its use for information resource description and discovery purposes as well as the description of agencies and services. Indeed the Schema is implementation neutral, defining no technological restrictions on how its elements are to be incorporated into systems, nor presuming any particular software architecture. It does not specify where, when or how metadata will be captured. The concern over time is that wherever, whenever and however metadata is captured, it will remain persistently linked to the record. Although metadata standards per se cannot guarantee such persistent associations, they can clearly demonstrate that assuring such persistence is an implementation imperative.

Implementors of the RKMS are enabled to identify and exploit a variety of technologies to populate the RKMS element fields. In a typical workplace, document management systems may be mandated to control the creation and dissemination of document level records, personnel systems mandated to map employees, their positions and their levels of authority, and workflow systems mapping information flows associated with business processes. Each of these aspects of seemingly disparate technologies are relevant to capturing specific metadata needed to produce reliable and authentic records over time.

The tendency in present records systems is to identify by user supplied tagging a variety of data elements which are then incorporated as contextual metadata around a document located in, or linked to, the records system. Such a response is appropriate where the risks of using parallel technologies to persistently associate metadata with the record are

judged to be too great. It is a common records-centric solution - if we cannot trust other systems to be sustainable over time, metadata cannot be merely associated with the record via pointers or links, but must be brought explicitly within the confines of the records system itself. This is the approach taken by the Victorian Electronic Records Strategy (VERS) project, amongst others. VERS defines metadata for specific levels of records aggregation, as well as specifying agent and business metadata to be associated with the record itself. Elements of the RKMS defined within the Records entity enable just such an approach. But whether this is the preferred solution depends on the circumstances of the implementation.

Figure 4: Coverage of Recordkeeping Metadata

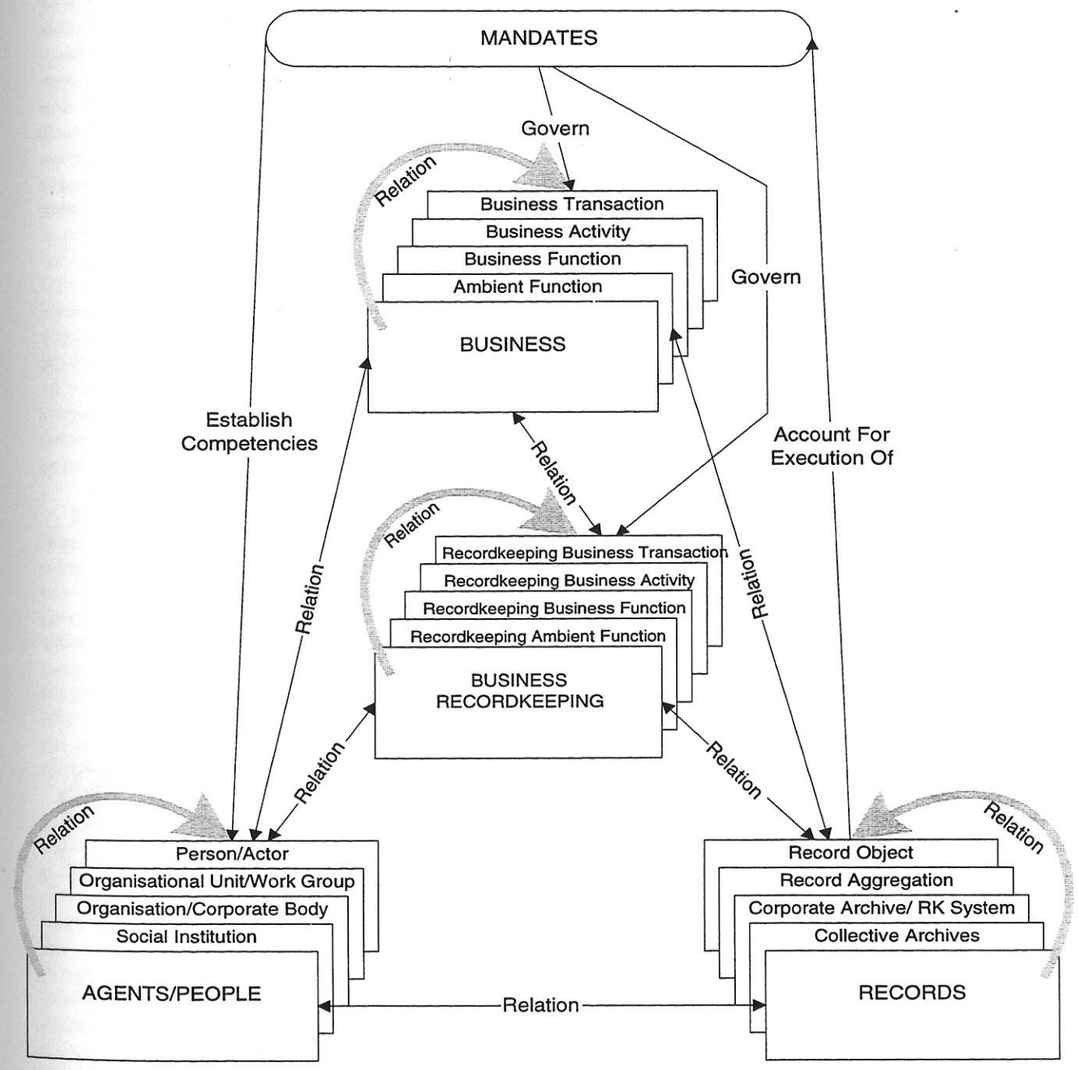
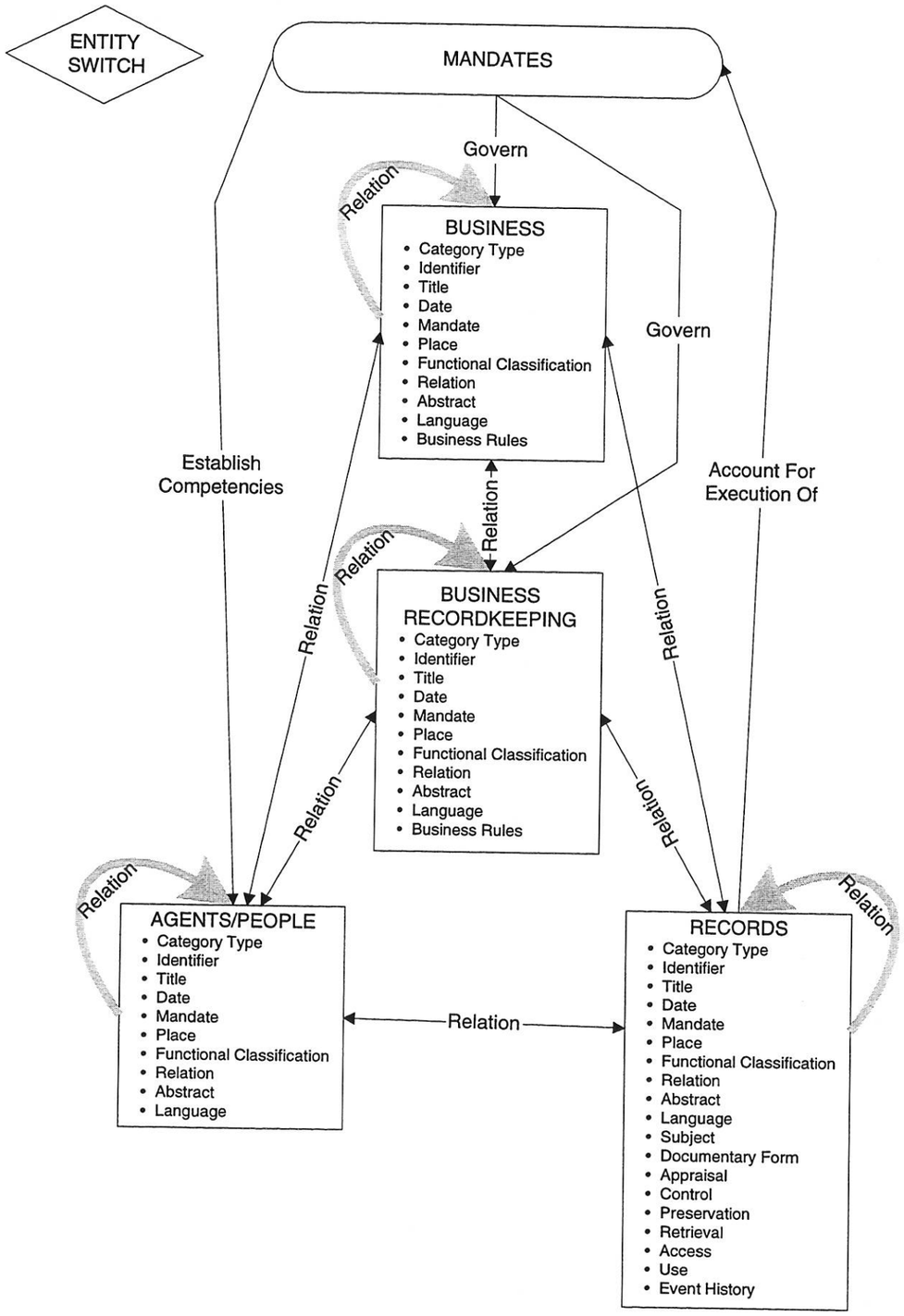


Figure 5: Recordkeeping Metadata Elements



While a variety of tags may be automatically attributed, a significant proportion cannot. An alternative approach would be to overlay the production systems with mechanisms to grab the document as it transacts business (perhaps as it is communicated beyond specified work group boundaries). Such mechanisms would associate the record with metadata from the document management system and any workflow or knowledge management systems engaged in the business process, as well as with data from the personnel system documenting the creator. This associated metadata would include system descriptions and dependencies. The associations between the record and its contextual metadata may be made by direct links into the nominated systems, by creation of metadata around the record created in robust formats lodged as discrete items into a storage location, or by embedding the metadata within the record[17]. As a possible means of 'future-proofing' records, these latter approaches appeal to recordkeepers, especially as the complexity of managing associations and links to disparate systems over time is, and may prove to be, an unsustainable burden for current technologies and the organisations that support them.

Such potential implementations begin to empower recordkeepers to connect with the newly emerging computer paradigms of component programming and non-proprietary, process specific program functionality. What we need to achieve are records which contain or are associated with all requisite metadata (from wherever it may be found), which are sustainable over time and over distributed network spaces. Alternative implementation strategies like this are envisioned by the RKMS, which looks ultimately to the concept of self-managing objects.

The Recordkeeping Metadata Schema encompasses more than documenting the immediate circumstances of creation. In implementing the Schema, organisations can determine the extent of the reach of their systems. If, for example, the records are only of relevance to a discrete organisational group, located within one area, the metadata may be minimal, as we can assume that contextual organisational knowledge will be implied. If a record's reach is beyond the organisation - as increasingly more documents are in distributed networks, with transactions enabled on the web via documentary carriers - then additional metadata which specifies these organisational parameters would need to be available to a wider audience to facilitate interpretation. The RKMS envisages scalable definitions of reach to be identified and configured into individual implementations through its layers of aggregation of organisation, business and record.

Defining the reach and the comprehensiveness of specific implementations will clarify for organisations the extent to which some or all of the elements are introduced and the ways in which the records created by business need to be 'bound' with metadata.

Conclusion

The RKMS uses recordkeeping understandings to make explicit connections between business, people who do business and the records which occur as a result of doing that business. It embraces traditional articulations of recordkeeping and enables future articulations. Much of the metadata work undertaken so far in electronic networked environments has been based on a passive notion of document-like information objects. The records and archives metadata community in Australia takes a different perspective in relation to records, regarding them as active participants in business processes and technologies rather than passive objects to be described retrospectively. Envisaging records as potentially self-managing information objects that act as the transactors of business has informed the SPIRT Recordkeeping Metadata Research Project. This vision links the dynamic world of business activity to the passive world of information resource in cyberspace.

The recordkeeping metadata approach is geared to implementation in an electronic environment in which doing business electronically and delivering services online is rapidly evolving. To be able to rely on the electronic business transactions which are, according to our politicians, our future, understandings of how to ensure these transactions are reliable and robust must be built into the new enabling technologies in an integral way. We come to the electronic business table with a firm proposal for incorporation into that agenda, one which is practical and implementable in a variety of ways. The RKMS is a tool for all players concerned with authoritative and reliable documentation that provides evidence of business transactions in electronic environments.

References

[1] Bearman, David and Sochats, Ken. 'Metadata Requirements for Evidence', draft paper dated October 1995. Paper accessed via http://www.oclc.org:5046/conferences/metadata/requirements.txt on 2 September 1998.

[2] Reed, Barbara. 'Metadata: Core Record or Core Business?', *Archives and Manuscripts*, Volume 25 November 1997 No 2, pp 218-242

[3] Weibel, Stuart. 'The State of the Dublin Core', in *D-Lib Magazine*, Volume 5 Number 4, April 1999. Paper accessed via http://www.dlib.org/dlib/april99/04weibel.html on 16 April 1999.

[4] Public Record Office Victoria, Ernst and Young and CSIRO. *Victorian Electronic Records Strategy Final Report 1999*, Public Record Office Victoria, 1998

Acknowledgements

We acknowledge the contribution of other Research Team members and associates, particularly Kate Cumming, APA(I) scholarship holder, Dr Nigel Ward and Dr Linda Bird, DSTC, and Geoff Acland-Bell (for the expert rendering of the conceptual models). We also acknowledge the input of members of the Project Steering Committee and our Expert Reference Group of Australian and international reviewers.

1 The Project involved collaboration between Chief Investigators Sue McKemmish, Monash University, and Ann Pederson,UNSW, and their industry partners from the National Archives of Australia, State Records Authority of NSW, Queensland State Archives, the Records Management Association of Australia and the Australian Council of Archives.

2 The acronym SPIRT derives from the name of the Research Grant which funded the Project, Strategic Partnership with Industry – Research & Training (SPIRT) Support Grant, which provides for joint funding by the Australian Research Council and the Industry partners.

3 Recordkeeping metadata is defined broadly to include all standardised information that identifies, authenticates, describes, manages and makes accessible documents created in the context of social and business activity. Recordkeeping metadata so defined has traditionally been captured and managed in both recordkeeping systems and archival control systems.

4 Within the archival community the ACA/ASA Committee on Descriptive Standards has endorsed the SPIRT Recordkeeping Metadata Schema as a framework for the Committee's future work on the development of domain specific recordkeeping metadata and archival descriptive standards. The Chair of this Committee has recently approached Standards Australia with a proposal to develop the SPIRT Recordkeeping Metadata Schema into a Framework Australian Standard for Recordkeeping Metadata.

5 The term "Schema" (plural schemata) is used to mean the semantic and structural definition of the metadata used to describe recordkeeping entities. A schema describes the names of metadata elements, how they are structured, their meaning and so on. The metadata community also refers to metadata schemata as metadata sets or specifications.

6 The Resource Description Framework (RDF) was developed by the World-Wide Web Consortium (W3C) to provide the foundation for metadata interoperability across different resource description communities, see: http://www.w3.org/RDF.

7 Object Role Modelling (ORM) takes a conceptual modelling approach that views the world in terms of objects and the roles they play. It is very expressive, enabling a high level of detail and rigorous analysis, and can be populated with data instances which thus allows for grounded validation.

8 For background information on the project, see Sue McKemmish, Adrian Cunningham and Dagmar Parer, 'Metadata Mania: Use of Metadata for Electronic Recordkeeping and Online Resource Discovery' in Place, Interface and Cyberspace: Archives at the Edge, Proceedings of the 1998 Conference of the Australian Society of Archivists, Fremantle 6-8 August 1998. Canberra. Australian Society of Archivists. 1999, pp129-144; and Sue McKemmish and Glenda Acland. 'Accessing Essential Evidence on the Web: Towards an Australian Recordkeeping Metadata Standard.' Paper for AusWeb99 Conference. Available at: http://ausweb99.scu.edu.au/aw99/papers/mckemmish. For details of project outcomes, visit the project web site at http://www.sims.monash.edu.au/rcrg/research/spirt/index.html

9 Information on the Investing for Growth strategy can be found at: http://www.dist.gov.au/growth/html/infoage.html

10 For further information see: http://law.gov.au/ecommerce/interim3.html

11 The Project Team, in developing a simple but high level framework model for recordkeeping metadata given as Figure 1, used as an example of effective visual representation the INDECS Community's "Model for Commerce" as derived from David Bearman, Eric Miller, Godfrey Rust, Jennifer Trant and Stuart Weibel, 'A Common Model to Support Interoperable Metadata: Progress report on reconciling metadata requirements from the Dublin Core and INDECS/DOI Communities.' D-Lib, Vol.5 No.1, January 1999. Available at: http://www.dlib.org/dlib/january99/bearman/01bearman.html

12 For details of the Australian Government Locator Service see http://www.naa.gov.au/govserv/agls/)

13 The Australian Government Locator Service (AGLS) Manual for Users, Version 1.1: 1999-06-09,' Office of Government Online, National Archives of Australia provides details of the application of these types of qualifiers - see http://www.naa.gov.au/govserv/agls/

14 Simon Cox has written an excellent discussion paper for the DC community on issues relating to structure, authority and qualification in DC: http://wwww.agcrc.csiro.au/projects/3018CO/metadata/dc-guide

15 The Business Acceptable Communications model is described at http://www.sis.pitt.edu/~nhprc/meta96.html

16 See http://www.naa.gov.au/govserv/techpub/rkms/intro.htm

17 The interactions of the Monash University, Archives and Records Faculty team (Sue McKemmish, Frank Upward, Barbara Reed and Chris Hurley) with David Bearman, particularly at the Managing the Records Continuum intensive workshops in Melbourne and Canberra in June 1996 informed the ideas presented here.

Effective Reuse of Textual Documents Containing Tabular Information

L.E.Hodge, W.A.Gray and N.J. Fiddian

Department of Computer Science,
Cardiff University,
Cardiff, UK.

{scmleh|wag|njf}@cs.cf.ac.uk

Abstract.

This paper presents an overview of a toolkit that can facilitate efficient reuse of tables appearing within textual documents [1]. In order to effectively reuse information contained in these documents, it is important that we process the accompanying text as well as any tables that appear. From this text, it may be possible to extract metadata such as descriptions of table content and related formulae, mappings and constraints. This metadata can then be exploited to enhance the value of extracted tables during their subsequent reuse. In this paper we present a discussion of the techniques used to process tables and associated text, both of which rely heavily on the use of regular expressions. Our techniques for locating tables utilise similar visual clues used by other table processing techniques discussed in the literature [5,6,7,8], although our approach to exploiting them is quite different. Our tools have been designed to provide a high level of support for the numerous types of table layout encountered in plain text tables, an area that has previously been somewhat overlooked.

1. Introduction.

The growth of the Internet has provided increased access to large bases of textual information that has potential for reuse. Of these documents, some of the most useful (in terms of reuse in information systems) are those that contain information in the form of tables. Tables offer a powerful mechanism for presenting related information in a clear and concise manner. The very nature of tables makes them suitable for reuse since it should be (relatively) straightforward to convert a table from a document into an equivalent table in a spreadsheet or database to facilitate integration with other information systems. Also, the fact that an author has gone to the trouble to present information in a tabular form, would indicate that they felt there was value in presenting the information in this way. This indicates to some extent that the content of the table may have value that can be reused.

In this paper, we focus our discussion on the processing of documents in the form of plain text files (i.e. those that contain only ASCII text[1]) as these are the most difficult type of human readable source document to process. Previous work on processing tables in plain text documents has resulted in a number of successful techniques for locating and processing tables. Unfortunately correct understanding of plain text tables has not always been possible using these techniques due to the large variety of layouts that can occur [2]. To overcome these problems, we have developed techniques that utilise the positions of components within the table to determine correct table layout.

Although previously, work has been undertaken to process tables that appear in textual documents, the textual component itself has been ignored. We feel that the text that accompanies tables can be useful in reuse as it often contains information relating to the content of the tables. In [3] we discuss the type of information that may be present in the text along with techniques to faciliate its location and extraction. This information can be employed as a form of metadata to augment any extracted tables and has two basic forms; textual descriptions of table content and technical descriptions i.e. formulae, mappings and constraints.

In this paper, we discuss the techniques we have developed to provide effective processing of both the table and the accompanying text.

[1] In our toolkit, we also support processing of documents that contain embedded mark-up such as HTML and Latex. Processing here is simplified by exploiting embedded tags.

2. Techniques for Processing Tables in Textual Documents.

Most of the techniques for the processing of tables in textual documents that are described in the literature, perform two stages of processing:

i) Locate tables within a document.
ii) Determine the positions of columns and rows within the table and exploit this to determine the layout of the table.

In our toolkit, we add two further processing stages:

iii) Process table content to determine the type of plain text layout occurring - a more powerful technique to determine the layout of the table.
iv) Process accompanying text to locate and extract any available metadata.

In this section, we will examine each of these processing steps in turn.

2.1 Techniques for Locating Tables.

When reading a document, a human locates tables based on their visual form. Computerised processing emulates human behaviour by exploiting the same types of visual clues. The most useful types of visual clue are:

i) Horizontal blocks of white space – commonly used to indicate the boundaries between different components of a document.
ii) Vertical blocks of white space – columns of white space occur between each column of entries in the table and are a good indicator of the presence of a table (see Figure 1).
iii) Horizontal rules – these can also be used to indicate the presence of tables as they are sometimes used to indicate the boundary between the labels and the rest of the table.
iv) Vertical rules – in a fully typeset table, vertical rules may also be used to indicate the edges and column boundaries of a table. Presence of vertical rules is uncommon as they are not essential to understanding and increase the cost of setting a table [2].

```
Name    Age     Sex
--------------------
Sarah   22      Female
Tim     23      Male
Jo      22      Female
Jon     24      Male
```

Figure 1: A simple plain text table.

Since the inclusion of rules is not mandatory, horizontal and vertical blocks of white space are the only reliable clues in the location of tables. In a number of previous systems [5,6,7,8], a combination of these clues has been used to enable location of tables. Basically, the source document is first broken into blocks by splitting where horizontal blocks of white space appear. Each of these blocks is then processed to check for blocks of vertical white space that could indicate column structure and thus, the presence of a table.

In our system, although we utilise both of these clues, we take a slightly different approach to processing. Initially, we don't split the document and process each block to check if it contains a table. Instead we process the document as a whole and potential tables emerge. We then use the positions of blank lines to check the boundaries of potential tables. Also, we don't perform complex processing on blocks of white space as used in other techniques, but instead employ more simple heuristics based on regular expressions [4] to increase efficiency.

To locate tables, we utilise two different types of heuristic:

i) Double (or more) space count.
ii) Type classification.

Our system utilises two heuristics to increase recall. Each of the heuristics is best suited to locating a certain class of table types (although both fare reasonably well on their own). Using both heuristics enables our system to locate most, if not all of the tables within a document quickly and efficiently.

Let us now discuss each of these heuristics in turn.

Double (or More) Space Count Heuristic.

This heuristic uses the idea that the columns in most tables are separated by at least two spaces. Thus by searching for such patterns in the text, it should be possible to locate the gaps between columns without extensive processing. We use the regular expression `\b\s\s+\b` to match these inter-column gaps[2]. For each line of the source document, we store a count of occurrences of two (or more) spaces. Using these counts, it is possible to locate the positions of potential tables. To do this, a threshold level is set (e.g. the average of the counts) and blocks of (two or more) consecutive lines whose count exceeds this threshold are marked as potential tables. Figure 2 illustrates how tables can be located in this way. The initial block located by this process is illustrated by the dotted rectangle. Notice that if the labels are separated from the body of the table by a horizontal rule (as they are here), the labels will not be included in the block defined by the initial location process. To overcome this problem, if a horizontal rule is located in the line above a potential table block, the top border of the block is moved in order to include the labels.

```
...they are not essential to understanding          0   1
and increase the cost of setting a table [2]        0   2
                                                    0   0
     Name      Age      Sex                         2   1
     --------------------                           0   0
    | Sarah     22       Female |                   2   2
    | Tim       23       Male   |                   2   2
    | Jo        22       Female |                   2   2
    | Jon       24       Male   |                   2   2
                                                    0   0
                                                    0   0
                                                    0   0
Figure 1: A simple plain text table.                0   2
                                                    0   0
Since the use of rules is not mandatory,            0   1
horizontal and vertical blocks of white             0   1
space are the only...                               0   1

Average Double Space Count = 0.534
Average Type Count =  1.134
```

Double Space Count | Type Count

Figure 2: Illustration of our table location heuristics.

The Type Classification Heuristic.

This technique exploits the power of regular expressions to act as classifiers for data types. For each line of the source table, we employ a set of regular expressions to divide it into tokens and classify them based on their content (see Figure 3). We support classifiers for string[3], integer, real, float, times and dates of various formats and monies with different currencies.

[2] In plain text, a full stop in a passage is usually followed by two spaces. To avoid matching these we use the \b meta-character in the regular expression. Thus reducing the number of matches found by only matching at word boundaries.
[3] To aid processing any stream of letters, spaces and punctuation is classified as a single string.


```
Name   No.   Balance     string
Dave   123   123.45      string,int,real
Fred   134   123.56      string,int,real
John   145   323.68      string,int,real
Carl   166   287.78      string,int,real
```

Figure 3: A table with associated type classifications.

As with the double (or more) space heuristic, we store a count of the number of data types that appear in each row. Again, potential tables can be located by locating blocks of lines whose type count exceeds a threshold (as illustrated in Figure 2). Generally, this technique is best at locating entries in the main body of the table. Extra processing using clues such as horizontal rules and blank space is used to locate labels if they are available[4].

2.2 Techniques to Locate Column and Row Positions.

Once a table has been located, it must be processed in order to determine the positions of columns and rows. Clearly, the column structure is indicated by the presence of vertical blocks of white space (streams) between each column[5]. In order to determine column structure we must exploit these white streams. The most commonly used technique [5,6] is to count the number of white space characters appearing in each column of the text. Peak values will appear where white streams exist and can indicate the positions of column separators. A number of other techniques, mostly based on bounding boxes have also been discussed [7,8].

For our tools, we have developed a technique based on exploiting white streams that is more complex than those commonly used. Instead of simply counting the frequency of space characters in each column of the text, our technique searches (and stores the size) for the longest continuous stream of white spaces that appear in each column of characters in the block resulting from the table location process. This length count is augmented by a weighting based on the number of non-white space characters that appear in the next column of characters. This indicates the number of entries that will appear in the column generated by this column separator and the next. If relatively few entries will appear in this column, it may not be valid. Thus, it may be incorrect to insert a column separator at this position.

[4] Again the presence of a horizontal line separates the labels from the body of the table. As before, the top border of the block is moved in order to include the labels.
[5] In some situations it is possible to exploit vertical rules that appear between columns. As these occur infrequently they will not be discussed here.

To locate the positions of column separators, we examine the maximum runs for each column of characters and extract the positions of those that have peak vales[6]. These give an indication of the possible positions of column separators and the weighting acts as an indication of the likelihood that a valid separator has been located.

Consider Figure 4 (an MS-DOS directory listing). If we have a threshold based on the average weighting, we can assign with confidence, four of the six column separators (i.e. those whose weighting exceeds this threshold). Confidence in the remaining separators is low since they don't fit in with the general format of the table. Clearly, to define a column separator at the position of the ninth character in each row would be incorrect as this would split the file names. Even so, we would wish to define a column separator at position 23 in order to define a column for file sizes. To enable this type of processing, two processing modes are required, fully automatic, where all deduced column separators are defined, or semiautomatic where those whose weighting exceed the threshold are automatically set and user interaction is required to decide the validity of any that remain.

To deduce row structure a similar technique is employed to locate horizontal white streams. Because of the complexity of table layout in plain text documents, it is not always possible to determine row structure by exploiting white space alone. To correctly deduce the layout, the positions of the components within the table also need to be considered (as discussed in Section 2.3).

```
JAVA          |     | <DIR>     |     | 11/12/96 | 15:19 | Java
PFE           |     | <DIR>     |     | 12/12/96 | 14:50 | PFE
PERPRO        |     | <DIR>     |     | 20/07/98 | 14:34 | perpro
PERL          |     | <DIR>     |     | 09/01/97 | 18:10 | Perl
QUERY_~1      |     | <DIR>     |     | 12/03/97 | 14:57 | Query_parser
TABLEP~1      |     | <DIR>     |     | 30/10/97 | 10:48 | TableParser
CLASSI~1      |     | <DIR>     |     | 14/05/98 | 14:45 | classifiers
VBCLASS       |     | <DIR>     |     | 01/07/98 | 14:41 | vbclass
FREETEXT      |     | <DIR>     |     | 16/09/98 | 13:45 | freetext
INTEGR~1      |     | <DIR>     |     | 01/09/98 | 16:09 | integrate
PK            | TXT |           | 160 | 23/10/98 | 10:34 | pk.txt
PK            | $$$ |           |  97 | 19/10/98 | 14:13 | pk.$$$
TABLEM~1      |     | <DIR>     |     | 18/02/99 | 16:32 | tablemodel
LOCATOR       |     | <DIR>     |     | 06/05/99 | 10:20 | Locator

12345678901234567890123456789012345678901234567890123456789012345678901234567

      Max runs found at:
      Position 9 with weighting 2
        "    " 15 with weighting 12
        "    " 23 with weighting 2
        "    " 28 with weighting 14
        "    " 38 with weighting 14
        "    " 44 with weighting 14

      Average weighting is: 9.67
```

Figure 4. Illustration of technique for locating column separators.

[6] Where a number of neighbouring columns have the same peak value, the last one in the group is chosen.

2.3 Techniques for Processing Table Content.

In most of the existing table processing systems, the techniques used to determine the layout of data in the table are fairly simple. In most cases, only limited processing is undertaken to determine the formatting of rows (based again on white space). Determining column structure has been more extensively investigated and multiple layers of labelling are often supported. Unfortunately, in most cases, the layout of tables is assumed to be fairly simple. Alas, when working with plain text tables, this is not always the case. The flexibility of layout offered by plain text results in a large variety of different types of table format. Many of which are not suitable for direct reuse. Previously little emphasis has been placed on providing support for the different types of layout that occur. In our investigation we have focused on the processing that would be required to transform source tables into a relational form, as this is the most suitable form for reuse[7].

In our toolkit we have attempted to support as many different types of layout as possible. We classify these types of layout based on the components of the table as illustrated in Figure 5.

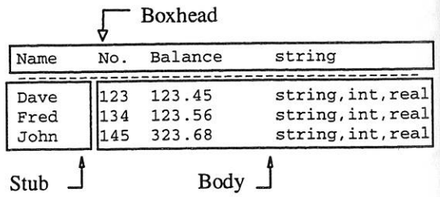


Figure 5: Components of a table.

Within each of these components we see different types of layout. The boxhead and the stub sections offer the author most control over the way that data is presented in the table and correct processing of these components is vital to correct reuse of table content[8]. For each component, let us now briefly describe the types of layout that occur.

Boxhead Layouts.

The simplest layout in the boxhead consists of a single layer of labels (see Figure 1). This type of boxhead is easy to handle, as no extra processing is required to reuse its content. The appearnce of multiple levels of labelling is also common in the boxhead (Figure 6). Here, grouping of related columns is possible, making the table content more self explanatory. Where multiple levels of labelling occur, the boxhead will need to be processed to deduce a single layer of labels. This is achieved by creating a simple compound label based on a downward traversal through the appropriate labels (i.e. `Examinations.Mid-term` and `Examinations.Finals`).

```
                Examinations
Name       Mid-term  Finals  Total_Mark  Grade
-----------------------------------------------

T.Thorne      70       71       70.5        A
S.Warrilow    65       70       67.5        B
J.Bloggs      64       78       71.0        A
```

Figure 6: A table with multiple layers of labelling in the boxhead.

In some cases (usually due to space constraints or appearance), labels may span multiple lines (Figure 7). Where this occurs it is important that our tools are able to determine the type of processing required and generate appropriate labels[9].

```
                    1900
                ------------------
                % of          Seats
Party           vote          won
xxxxxxx         xx            xxxxx
```

Figure 7: An example of a boxhead containing multi-level labelling.

Stub Layouts.

The simplest type of stub (as in Figure 6) requires no processing as it consists of a single column of entries. Stubs in plain text tables also support a type of classification hierarchy through the use of nested entries (Figure 8). This allows elimination of repeated labelling resulting in a more compact table. Where stubs of this kind are encountered, they must be expanded into multiple columns (one for each level of indent) and repeated entries reintroduced (Figure 9). Notice that stubs with nested entries rarely have a label in the boxhead. Thus, when a stub of this kind is expanded, appropriate labels will need to be inserted by the user.

```
                    Price
--------------------
Ford
  Fiesta
    1.1          8000
    1.3          8650

Vauxhall
  Astra
    1.3          8500
    1.6          9250
    2.0          12000
```

Figure 8: A table with a stub containing nested entries.

```
Maker      Model    E_Size  Price
-------------------------------
Ford       Fiesta     1.1   8000
Ford       Fiesta     1.3   8650
Vauxhall   Astra      1.3   8500
Vauxhall   Astra      1.6   9250
Vauxhall   Astra      2.0   12000
```

Figure 9: The result of processing the table in Figure 8.

Again, stub entries may span over multiple lines. Where this occurs, it is important that the entry is handled as a single entity. Where multi-line stub entries occur, it is essential that some visual clues can be located to allow correct processing (e.g. blank lines or rules between each row).

Body Layouts.

The simplest body layout is identical to a relational table and requires no special processing (as in Figure 4). Problems can sometimes occur where entries in the body span multiple lines (Figure 10). Where this occurs, correct processing relies on visual clues as it does for multi-line stub entries.

```
Model          Description
-----------------------------------------
Pro-Yo II      Wooden axle, beginner
               or intermediate yoyo

RB II          Roller bearing transaxle
               yoyo.  Good looper.
               intermediate / Expert use.

Viper          Roller bearing transaxle
               yoyo.  Butterfly shape
               good for string tricks.
               Rubber sides !
```

Figure 10: A table with multi-line entries in the body.

[7] Relational tables can be easily imported into most databases and spreadsheets. Techniques to support integration with other types of DBMS are well understood.

[8] The layout of the table body also affects the processing, although not to the same degree as the boxhead and stub.

[9] Due to space constraints we are unable to discuss processing of multi-line entries in this paper.

2.4 Techniques for Extracting Metadata from Accompanying Text.

From the text that accompanies tables, we support the extraction of four types of metadata:

i) Semantic descriptions – textual descriptions related to the content of columns in the table.

ii) Formulae – that define how a column may be materialised from others e.g. Total_Mark = (Mid-term + Finals)/2.

iii) Mappings – that define how a column may be materialised from others e.g. if Total_Mark >= 70 then Grade = 'A'.

iv) Constraints – that define the range of valid entries for a column e.g. Total_Mark >= 0 and <=100.

For all types of metadata our extraction techniques utilise regular expressions. In simple terms, processing involves dividing the text into sentences, then checking to see if any metadata appears in these sentences by matching using a regular expression.

Regular expression to match a formula.

```
(\w*\s*=\s*((\(*|\)*)\s*)?(-?([0-9]+(\.
[0-9]*)?|\.[09]+)|\w*)\s*((\(*|\)*)\s*)?
(\+|-|\/|\*|\^)\s*((\(|\))\s*)?(-?([0-
9]+(\.[0-9]*)?|\.[0-9]+)|\w*)((\s*((\
(|\))\s*)?(?([0-9]+(\.[0-9]*)?|\.[0-9]+)
|\w*)(\s*((\(|\))\s*)?(\+||\/|\*|\^)\s*
((\(|\))\s*)?\w*(\s*(\(|\))\s*)?)*)?)(\s
*(\(|\))\s*)?)
```

Regular expression to match a mapping .

```
(if(\s*)\w*(\s*)?(\<|\>|=|\<\>|\<=|\>=)(
\s*)\d*(\s*)?(and(\s*)?(\<|\>|=|\<\>|\<=
\>=)(\s*)?\d*(\s*)?)?then(\s*)?\w*(\s*)?
=\s*((\(*|\)*)\s*)?(-?([0-9]+(\.[0-]*)?
|\.[0-9]+)|\w*)\s*((\(*|\)*)\s*)?(\+|-
|\/|\*)\s*((\(|\))\s*)?(-?([0-9]+(\.[0-
9]*)?|\.[0-9]+)|\w*)((\s*((\(|\))\s*)?(-
?([0-9]+(\.[0-9]*)?|\.[0-9]+)|\w*)(\s*
((\(|\))\s*)?(\+||\/|\*)\s*((\(|\))\s*)?
\w*(\s*(\(|\))\s*)?)*)?)(\s*(\(|\))\s*)?
)
```

Regular expression to match a constraint .

```
(\w*(\s*)?(\<|\>|=|\<\>|\<=|\>=)(\s*)\d*
(\s*)?and(\s*)?(\<|\>|=|\<\>|\<=|\>=)(\s
*)?\d*)
```

Figure 11: The regular expressions used to locate formulae, mappings and constraints.

For semantic descriptions this simply involves searching for column labels in the text. If they appear, the appropriate sentences are extracted and are used to build a type of key or dictionary. This can then be used by the end user to get a quick overview of the content and meaning of the table without reading all of the accompanying text. For the more technical metadata, we utilise the regular expressions shown in Figure 11. Where successful matches are made, this metadata is stored for reuse in the destination application.

3. Reuse of Table Content and Metadata.

In order to demonstrate how useful content extracted from documents can be reused, our toolkit provides facilities to export tables and associated metadata into two types of application – an Excel spreadsheet and an INGRES database.

We support reuse in Excel via the Visual Basic for Applications API. This allows our tools to exploit the functionality of Excel to generate appropriate spreadsheets with appropriate formulae, mappings and constraints. Semantic descriptions are used to generate pop up descriptions attached to the label for each column, allowing the user a brief description of the content of each column.

To support generation of an INGRES database, our tools generate appropriate SQL to set up and populate a table[10] along with appropriate rules and procedures to support formulae, mappings and constraints. Obviously we can not utilise semantic descriptions directly in INGRES so these are made available to the user in a text file.

4. Conclusion.

In this paper we have presented an overview of a toolkit to facilitate the effective reuse of textual documents that contain data in a tabular form. Whereas previous table processing systems have concentrated solely on the location and processing of tables, our tools exploit the accompanying text to extract any metadata that may be available. This metadata is then used to add value to the extracted tables.

We have specifically designed tools to support the numerous different table layouts that are available in plain text documents. By developing techniques for the different types of layout that occur in different table components we are able to achieve a high level of precision and recall.

Although in this paper, we have concentrated our discussion on plain text source documents, our toolkit also supports textual documents that contain embedded mark-up, such as HTML and Latex. In providing support for documents of this type (i.e. document definition languages DDL's) we have utilised a meta-language approach [9]. Using this approach allows simple extension of our processing to support any additional DDL's should the need arise.

[10] We use a system or regular expressions to determine the data type for each column in order to define the table schema.

References.

[1] L. E. Hodge, W. A. Gray and N. J.Fiddian. A Toolkit to Facilitate the Querying and Integration of Tabular data from Semi-structured Documents. In *Proceedings of the 16th British National Conference on Databases, (BNCOD 16)*, July 1998.

[2] *The Chicago Manual of Style*, Thirteenth Edition, The University of Chicago Press, 1982.

[3] L. E. Hodge, W. A. Gray and N. J. Fiddian. Deduction of Metadata for Tabular Structures in Plain Text. In *Proceedings of IEE Conference on Multimedia Databases and MPEG-7*, 1999.

[4] J. Friedl. *Mastering Regular Expressions*, O'Reilly and Associates, 1997.

[5] S. Chandran and R. Kasturi. Structural Recognition of Tabulated Data. In *Proceedings of the International Conference on Document Processing (ICDAR 93)*, 1993.

[6] K. Itonori. Table Structure Recognition Based on Textblock Arrangement and Ruled Line Position. In *Proceedings of the International Conference on Document Processing (ICDAR 93)*, 1993.

[7] M. A. Rahgozar, Z. Fan and E. V. Rainero. Tabular Document Recognition. In *Proceedings of the SPIE Conference on Document Recognition*, 1994.

[8] T. G. Kieninger. Table Structure Recognition Based on Robust Block Segmentation. In *Proceedings of Electronic Imaging 98 (SPIE), Document Recognition*, 1998.

[9] D. I. Howells. *A Source-to-Source Meta-translation System for Database Query Languages.* Ph.D. Thesis, Cardiff University, 1988.

Transformation-Based Learning for Automatic Translation from HTML to XML

James R. Curran
Basser Department of Computer Science
University of Sydney
N.S.W. 2006, Australia
jcurran@cs.usyd.edu.au

Raymond K. Wong
Basser Department of Computer Science
University of Sydney
N.S.W. 2006, Australia
wong@cs.usyd.edu.au

Abstract

Format tags implicitly represent content information in the same ambiguous, context dependent manner that words represent semantics in natural language. Translation from format to content markup shares many characteristics with tagging and parsing tasks in computational linguistics. The transformation-based learning (TBL) paradigm has recently been applied to numerous computational linguistics tasks with considerable success. We present a transformation-based translator which automatically learns to translate semistructured HTML documents formatted with a particular style to XML using a small set of training examples.

Keywords HTML, XML, markup, document processing, machine translation, machine learning

1 Introduction

XML [1] is the focus of increasing interest in markup representing *content* rather than *form*. Content tags describe the structural semantics of a document explicitly while form tags represent semantics implicitly as a function of document formatting. For instance, the XML tag `<TITLE>` describes a title, whereas the HTML tag `<B>` describes a bold font which may represent a title or one of many other document structures.

The XML revolution is resulting in huge quantities of legacy format tagged documentation in HTML, postscript, and word processor files which will need to be converted. There is also a need to convert between XML languages and recover the XML from XML documents formatted using XSL. Automatic methods for translation from format to content tagged documents are therefore of considerable research and commercial value.

The structural semantics represented by formatting are ambiguous, and context dependent, like the representation of semantics by natural language. For instance, Part of Speech (POS) tagging describes the role that each word plays in a sentence, such as noun (`NN`) or adjective (`JJ`)[1]. Each tag is co-dependent on the surrounding words and their tags. The TBL paradigm, proposed in [4], has been successfully applied to POS tagging [2, 5, 6]. Other TBL applications include text segmentation and classification [11, 9] and prepositional phrase attachment [7] which all involve learning annotations with extent which is necessary for HTML to XML translation. This work extends our TBL linear format to content tag translation, [8], where format and content tags are flattened not nested.

2 Transformation-Based Learning

Transformations are rules that are applied in a given context to change a tag from an erroneous value to its correct value. TBL automatically acquires a sequence of transformations to incrementally patch tagging errors. Transformations are applied to a simple initial guess at the correct tag. The POS tagger [2] initial guess was the most common tag for each word which was extracted from the training corpus. For TBL syntactic parsing [3], a right branching bracket structure, which is most common in English, was used as the initial state. The initial state does not need to use domain dependent knowledge and may represent no useful information such as the `NULL` tag.

A transformation modifies a tag when the context (such as neighbouring tags or words) of the tag matches the context described by the transformation. Every transformation is applied in sequence to every tag site in the document. The first ten POS tagging transformations from [2] are shown in Figure 1.

The transformation learning algorithm is illustrated in Figure 2. The training corpus is first stripped of the annotations that the tagger will learn. This stripped corpus is then annotated with the naive initial tagging which has been derived from statistical analysis of the training corpus or limited domain knowledge. The TBL algorithm

[1] Appendix A describes each POS tag

Proceedings of the Fourth Australasian Document Computing Symposium, Coffs Harbour, Australia, December 3, 1999.

#	tag	to	when
1	NN	VB	prev. tag is TO
2	VBP	VB	one of the prev. 3 tags is MD
3	NN	VB	one of the prev. 2 tags is MD
4	VB	NN	one of the prev. 2 tags is DT
5	VBD	VBN	one of the prev. 3 tags is VBZ
6	VBN	VBD	prev. tag is PRP
7	VBN	VBD	prev. tag is NNP
8	VBD	VBN	prev. tag is VBD
9	VBP	VB	prev. tag is TO
10	POS	VBZ	prev. tag is PRP

Figure 1: Transformations for POS tagging [2]

attempts to duplicate the training corpus from the initial tag state by iteratively learning transformations which patch errors in the current tag state. In each iteration, shown highlighted in Figure 2, the best scoring transformation in the current state is learned. From the current state, training corpus and transformation templates, a set of possible transformations is proposed. The error driven model uses each erroneous tag context to propose transformations by template instantiation. Each transformation is then evaluated using the training corpus and evaluation function. In the POS tagger [2], the evaluation function was the net accuracy improvement as a result of applying the transformation. The highest scoring transformation is appended to the learnt sequence and applied to the current state. Iterations continue until the desired tagging accuracy has been reached or none of the proposed transformations have a positive score.

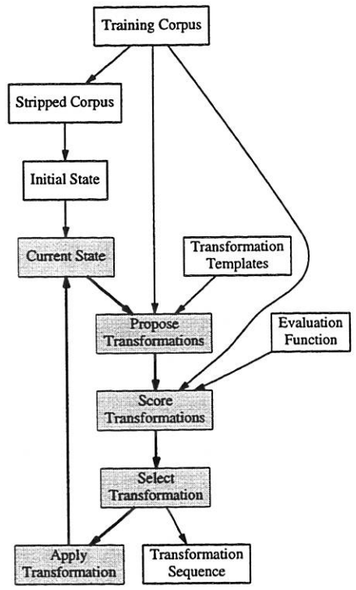


Figure 2: The transformation learning process

The transformation search space is limited by transformation templates. Templates describe valid transformations and must describe properties that will reliably indicate when a transformation is applicable. If a context relationship is not described by a transformation template then it cannot be represented by the system. Proposed transformations are formed using templates instantiated by filling their slots with the context of the erroneous tags. The POS tagging transformation templates from [5], shown in Figure 3, are primarily simple bigram and trigram relationships between neighbouring words and tags.

Change tag A to tag B when:

1. the *preceding* (*following*) word is tagged Z
2. the word *two before* (*after*) is tagged Z
3. *one* of the *two preceding* (*following*) words is tagged Z
4. *one* of the *three preceding* (*following*) words is tagged Z
5. the *preceding* word is tagged Z and the *following*) word is tagged Y
6. the *preceding* (*following*) word is tagged Z and the word *two before* (*after*) is tagged Y

Figure 3: Templates for POS tagging [5]

3 HTML to XML

A transformation-based learning system is defined by the:

- *initial state* that is applied to the document to be tagged
- *evaluation function* that is used during training to evaluate the quality of the proposed transformations
- *set of transformation templates* that define the context relationships that learned transformations can represent

We define these system parameters for HTML to XML translation in the following sections but first discuss preprocessing of the HTML and XML files.

A typical problem is the translation of a list of published papers on an academic's web page into XML (or bibtex) so that it can be loaded into an object database of research papers. Figure 4 and Figure 5 are fragments of HTML and XML that may be part of the training and target data for this problem. We will use examples from these fragments as motivation for the design of our system.


```
<H1><A NAME="SECTION00040000000000000000"></A>
<IMG SRC="OLD/blueball.gif" HEIGHT=14 WIDTH=14 ALIGN=BOTTOM>
<FONT COLOR="#00FFFF">Publications</FONT></H1>
<FONT COLOR="#0000FF"><FONT SIZE=-2></FONT></FONT>
<UL>
<LI><B>R.K. Wong</B>, H.L. Chau, and F.H. Lochovsky.
<FONT COLOR="#FFFF00">Dynamic Knowledge Representation in DOOR.</FONT>
In IEEE KDEX Workshop,Newport Beach, CA, November
1997.<A HREF="Papers/design.ps">(The full postscript (journal) version)</A> </LI>
<LI><B>R.K. Wong</B>, H.L. Chau, and F.H. Lochovsky.
<FONT COLOR="#FFFF00">A Data Model and Semantics of Objects with Dynamic Roles.</FONT>
<I>1997 IEEE International Conference on Data Engineering</I>, pp.402-411, UK,
April 1997.<A HREF="Papers/icde97.ps">(The postscript file)</A> </LI>
</UL>
```

Figure 4: A fragment of HTML from the Publications translation task

```
<Publications>
  <paper>
    <author> R.K. Wong </author>
    <author> H.L. Chau </author>
    <author> F.H. Lochovsky </author>
    <title> Dynamic Knowledge Representation in DOOR </title>
    <workshop> IEEE KDEX Workshop </workshop>
    <location> Newport Beach, CA </location>
     <date> November 1997 </date>
    <softcopy HREF="Papers/design.ps" />
  </paper>
  <paper>
    <author> R.K. Wong </author>
    <author> H.L. Chau </author>
    <author> F.H. Lochovsky </author>
    <title> A Data Model and Semantics of Objects with Dynamic Roles </title>
    <conference> IEEE International Conference on Data Engineering </conference>
    <location> UK </location>
    <pages> 402-411 </pages>
    <date> April 1997 </date>
    <softcopy HREF="Papers/icde97.ps" />
  </paper>
</Publications>
```

Figure 5: XML fragment corresponding to Figure 4

```
<H1>
<IMG HEIGHT=14 WIDTH=14 ALIGN=BOTTOM><SRC>OLD/blueball.gif</SRC></IMG>
<FONT COLOR="#00FFFF">Publications</FONT></H1>

<UL>
<LI><B>[R.K. Wong]</B>, [H.L. Chau], and [F.H. Lochovsky].
<FONT COLOR="#FFFF00">[Dynamic Knowledge Representation in DOOR].</FONT>
In [IEEE KDEX Workshop],[Newport Beach, CA], [November
1997].<A><HREF>[Papers/design.ps]</HREF>(The full postscript (journal) version)</A> </LI>
<LI><B>[R.K. Wong]</B>, [H.L. Chau], and [F.H. Lochovsky].
<FONT COLOR="#FFFF00">[A Data Model and Semantics of Objects with Dynamic Roles].</FONT>
<I>1997 [IEEE International Conference on Data Engineering]</I>, pp.[402-411], [UK],
[April 1997].<A><HREF>[Papers/icde97.ps]</HREF>(The postscript file)</A> </LI>
</UL>
```

Figure 6: Preprocessed HTML with aligned segments

3.1 Preprocessing

The transformation process begins by removing HTML tags that do not provide any useful information such formatting that does not apply to any text or targets within the text. In Figure 4, both line 4 and `<A NAME="SECTION00040000000000000000"></A>` can be removed without loss of information. Formatting is removed because transformation templates define contexts of limited length which means that significant HTML tags may be obscured by extraneous tags, which are commonly found in HTML generated by web page authoring software.

Images may be removed, though it is difficult to decide if an image (such as a ball used as a point marker) represents document structure. The dimensions of the image and frequency of use within the file may give some indications as to the purpose of the image. Further, removing images is dangerous because they may be part of the XML that is produced if they are illustrations. Using image information is outside the scope of our current system.

Some attributes such as `HREF` within `A` tags need to be extracted. For consistency, we convert these directly to a pseudo-HTML canonical form, with `HREF` as a child tag within the `A` markup. Only attributes that reference external entities such as URL's should be converted to the canonical form. Other attributes should remain in the form of `PROPERTY=VALUE`. Figure 6 shows the attributes `SRC` and `HREF` converted. Comparing URLs in the HTML and XML may allow more intelligent processing.

3.2 Alignment and Chunking

After preprocessing, the text within the HTML and XML documents is aligned. Alignment is performed in a single linear scan, since the XML document only contains a subset of the text (not tags) from the HTML document. After alignment, it is obvious that some words, phrases and punctuation have been discarded in the XML. Conjunctions, key words and punctuation in structured, regular text play a significant role in determining the content of a document. The non-aligned segments in Figure 4 are words `and`, `In`, `1997`, and `pp`, phrases in round brackets, and commas and periods. These words are often not aligned because the XML tags make them superfluous by defining the content directly. For this reason, we consider non-aligned words with significance to format tags. Keywords may also be found by searching for the XML tags in the HTML text itself. For instance, if the words `conference` or `workshop` appear frequently within the tags `<conference>` or `workshop` then they are probably significant words in determining the XML label.

Aligned text surrounded by XML tags is considered a text chunk in training. These chunks are marked using square brackets in Figure 6. Although this information is not available in target documents we can still utilise this segmentation to learn transformations as long as the transformation context does not refer to chunk boundaries. Using text segmentation tools, which should be fairly accurate with formatting information available, we may be able to include text chunks in the transformation contexts. This is particularly important for applications where formatting is sparse or does not cover information within the text itself that needs to be marked up. Phrase segmentation and identification have been implemented using the TBL paradigm [9, 11] and our system shares some common template types with these systems.

The text chunk boundaries identified using alignment with the XML training data form the skeleton XML tree structure. XML crystalises at the chunk boundaries where tags are added, substituted, removed, and reordered. Although transformations are learnt to manipulate the tags at the boundaries, they do not refer to these boundaries.

4 Initial State Tagging

The tagger will not know the boundaries of the text chunks where the XML tags should be attached, except when an accurate text segmentation tool has been applied first. For this reason, the initial state tagging cannot make a simple guess at the correct tag or its location, so a `NULL` initial state is used.

Using a good guess initial state has the advantage of reducing the number of transformations that a system is required to learn to reach the desired level of accuracy. Although there is no practical good guess for this system, the larger number of rules will not be much of a problem because the training sets are not large and the efficiency of the finite state transducer (FST) tagger implementation [10] is not dependent on the number of rules, although it does increase the size of the transducer.

5 Evaluation Function

Choice of the evaluation function is crucial to the quality and subtlety of the set of transformations learnt by the system. To acquire subtle transformations, the evaluation function must differentiate between small changes in the current state that indicate it is closer to the training corpus. To learn quality transformations, that can duplicate all structural aspects of the desired result, the evaluation function must be able to take into account all of these structural features.

The evaluation function for syntactic parsing [3] measures the correctness of a bracketing structure as the percentage of constituents (strings between matching brackets) from the current state that do not cross any constituents of the correct bracketing. This is not a problem in our system since we already have text chunks as a result of alignment. However, we still need to consider non-aligned words erroneously appearing within XML markup and the incorrect ordering of multiple markup tags at each chunk boundary.

The total evaluation score is the sum of the error function at every chunk boundary. The error function at each chunk boundary is:

1. for every missing opening or closing markup tag, add 10 to the error
2. for every extra opening or closing markup tag, add 10 to the error
3. for every non-aligned word that is within a markup pair, add 1 to the error
4. for every pair of tags that is in the wrong order, add 1 to the error

Figure 7: HTML to XML transformation evaluation function

Transformations are learned to minimise this evaluation function. The most errors will be patched by transformations that add the most frequent tags in the least ambiguous contexts. The most frequent tags primarily appear in the leaves of the XML tree structure which means that the XML is often constructed from the most nested layers outwards.

We will continue to experiment with the evaluation function in Figure 7 to improve transformation reliability and generality by taking other features into account, such as the tags each transformation refers to. For instance, the `<H1>` HTML tag will probably provide us with less ambiguous structural information, further up the tree structure than the `<I>` tag. There may be an advantage of starting from the least nested XML tags because there is less ambiguity and once tags are added to chunk boundaries they can be used in other transformations to further reduce the ambiguity. The training algorithm can be coerced into adding XML tags to the least nested chunks first by penalising transformations according to the depth of the XML they manipulate.

6 Transformation Templates

Transformation templates must describe various relationships between the elements of the current state which reliably indicate whether the transformation should be applied at that site. These elements include beginning and end markers, content and format tags, and words and punctuation within the HTML document. The possibly large number of format tags per content tag, may cause the format tags to obscure the content tags from transformations if considered equal. We have resolved this by expanding the proximity relationship. A content tag is *directly next* to a tag if it touches the tag and is *indirectly next* to a content tag if it is the first content tag within a given number of tags. The transformation templates are given in Figure 8.

insert ctag `X` when:

1. the (next,prev) (htag,naw) is `Y`
2. the (direct,indirect) (next,prev) ctag is `Y`
3. the (next,prev) word is `w`
4. the (next,prev) two (htag,ctag,naw) are `Y` and `Z`
5. the prev (htag,ctag,naw) is `Y` and the next (htag,ctag,naw) is `Z`

replace ctag `X` with (`Y`,`NULL`) when:

1. the (next,prev) htag is `Z`
2. the (direct,indirect) (next,prev) ctag is `Z`

remove (word,punctuation) `w` when:

1. the (next,prev) (htag,naw) is `Y`
2. the (direct,indirect) (next,prev) ctag is `Y`
3. the (next,prev) two (htag,ctag,naw) are `Y` and `Z`
4. the prev (htag,ctag,naw) is `Y` and the next (htag,ctag,naw) is `Z`

swap ctags X Y when: the (next,prev) (ctag,htag,naw) is `Z`.

Figure 8: Transformation templates for HTML to XML conversion. htag = HTML tag, xtag = XML tag, naw = non-aligned word or punctuation.

Most TBL systems have used a non-`NULL` initial state with transformations that substitute tags rather than add and delete them. Our system must manipulate multiple tags at each chunk boundary, and because of the empty initial state, must be able to change the number of tags. Transformations must be able to insert tags before, after and

between format and content tags, remove tags, puctuation and words and swap the order of content tags.

Our system has many more transformation templates than previous TBL systems. A large number of transformation templates means that the search space is large. However, since the size of the training data is much smaller than previous applications and the maximum number of errors in the current state is relatively small, a larger search space is not a problem.

7 Cleaning Up

Once all of the learned transformations have been applied in sequence, there are numerous XML tags embedded within the text and between the tags of the HTML. Transformations will have already removed the excess non-aligned text. The HTML tags are then removed leaving the XML document with XML tags and correct text chunks. Our current system does not deliberately balance begin and end XML tags. However, it is implicit in error minimisation that matching tags will be learned. The result must then be parsed with a DTD and corrected if necessary.

8 Conclusion

We have presented the first general, automatic translation system from HTML to XML based on Brill's transformation-based learning paradigm. Work in TBL POS tagging and other linguistic annotation tasks has contributed various techniques that can be applied to the transformation of HTML into XML. Initial processing is applied to the document to translate HTML tags into canonical form and to remove formatting tags that do not contribute any structural information.

A system based on the TBL paradigm is defined by the initial state tagging, error function and the set of transformation templates. We base our learning on alignment of the text chunks within the HTML and XML markup. The initial state tagging does not apply any tags. Tags are added to the beginning and end of each text chunk that is aligned between the HTML and XML training data. The error function evaluates the number of correct and incorrect tags and their order at each beginning and end of each text chunk and the number of correct words in each aligned text chunk. The transformation templates include simple templates to add/remove/reorder content tags depending on context of format and content tags, keywords and other words and punctuation within the training data. Words and punctuation that can be used as part of the segmentation process such as conjunctions and commas are automatically identified by the fact that they do not appear in the aligned XML data. Other transformation templates are used to change the size of the text chunks surrounded by each XML tag to remove unnecessary punctuation, conjunctions and extraneous words.

The final step of the process is to extract the HTML tags leaving just the XML and the text chunks in between. The HTML that is extracted from the XML may be used to generate an XML style file. Future work will involve investigating this possibility. The XML is then converted from the canonical form into the desired form using other simple transformations.

Our HTML to XML conversion system is very flexible and can cope with documents with extensive or limited structure. The tagger can be converted to an optimally efficient FST [10], and the learning algorithm requires only a small training set. This algorithm can be applied in a bootstrap manner: Initial training begins on a small manually tagged corpus; The tagger is then used to translate a larger corpus, which is then corrected and used to retrain the tagger. This train-correct cycle continues until the desired accuracy and coverage is achieved. This method greatly reduces the cost of manually tagging training data. Many legacy HTML documents will need to be migrated to XML, ensuring the continuing research and commercial interest in flexible automatic solutions. Used in conjunction with automatic linguistic tools, such as a POS taggers and shallow parsers, format conversion forms the basis of an automatic text summarisation and information retrieval system that generates XML directly. Future research will explore such possibilities.

References

[1] T. Bray, J. Paoli and C.M. Sperberg-McQueen (editors). *Extensible Markup Language (XML) 1.0.* Word Wide Web Consortium, 1998. REC-xml-19980210.

[2] E. Brill. A simple rule-based part of speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing*, Trento, Italy, 1992.

[3] E. Brill. Automatic grammar induction and parsing free text: A transformation-based approach. In *Proceedings of the 31st Meeting of the Association for Computational Linguistics*, Columbus, Oh., 1993.

[4] E. Brill. *A Corpus-Based Approach to Language Learning.* Ph.D. thesis, Department of Computer and Information Science, University of Pennsylvania, 1993.

[5] E. Brill. Some advances in transformation-based part of speech tagging. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, Seattle, Wa., 1994.

[6] E. Brill. Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. *Computational Linguistics*, 1995.

[7] E. Brill and P. Resnik. A transformation-based approach to prepositional phrase attachment disambiguation. In *Proceedings of the 15th International Conference on Computational Linguistics*, Kyoto, Japan, 1994.

[8] J.R. Curran and R.K. Wong. Transformation based learning in document format processing. In *Working notes of the AAAI 1999 Fall Syposium on Using Layout for the Generation, Understanding or Retrieval of Documents*, 1999.

[9] L. Ramshaw and M. Marcus. Text chunking using transformation-based learning. In *Proceedings of the Third ACL Workshop on Very Large Corpora*, 1995.

[10] E. Roche and Y. Schabes (editors). *Finite-State Language Processing.* The MIT Press, 1997.

[11] M. Vilian and D. Day. Finite-state phrase parsing by rule sequences. In *Proceedings of the 16th International Conference on Computational Linguistics*, 1996.

A Penn Treebank POS tags

Tag	Description	Examples
DT	Determiner	the, this, that
JJ	Adjective	recent, such, high
MD	Modal	should, can, must
NN	Noun	fact, house, weekend
NNP	Noun, proper	Michael, Sydney
POS	Possessive Ending	's, '
PRP	Pronoun	it, us, you
VB	Verb	be, make, have
VBD	Verb, past tense	was, said, placed
VBN	Verb, past participle	paid, named, made
VBP	Verb, non-3rd person, singular, present	do, argue, have
VBZ	Verb, 3rd person, singular, present	is, has

High Bit rate Wavelet Domain Digital Watermarking of Images and Compression Tolerance

Ashoka Jayawardena, Bob Murison and Patrick Lenders

School of Mathematical and Computer Science
University of New England
Armidale, N.S.W

E-mail {ashoka,rmurison,pat}@mcs.une.edu.au

Abstract

The increased commercial activity in internet and media industry demands protection of media such as images, video and audio against unauthorised processing and use. Watermarking is a technique to hide information in media such that the hidden information (watermark) is invisible. This hidden information can be a small sequence of bits resulting in low bit rate watermarking. In low bit rate watermarking, each information bit is represented by an invisible broadband signal when added to the image. The hidden information may be in the form of images in which case large number of bits must be embedded. Such watermarks are known as high bit rate watermarks and are usually binary images.

Unlike low bit rate watermarking where each information bit is transformed to a broadband signal by generating a suitable pseudo-random sequence, high bit rate watermarking demands some other means of embedding and detection due to large number of information bits to be embedded. We use a binary feature based watermarking technique on wavelet domain. Our work is inspired by the work in [2].

Our motivation for this research is twofold. Firstly, we embed the watermark in wavelet domain rather than DCT domain motivated by the fact that the wavelet transform is used in jpeg2000 standardisation process. Secondly, high bit rate watermarks tend to get destroyed, even before average lossless compression ratios between 20% to 32%. We wanted to get the watermark breakup point up to at least within 20% to 32%.

Keywords Watermarking, image compression, image authentication.

1 Introduction

Watermarking aims at achieving copyright protection. This can be done by visibly or invisibly adding information to the image [2, 10, 4, 9, 11, 8, 6, 5] or documents [7] to be protected. The functionality provided by the copyright protection scheme varies depending on the application. The copyright can be a message which carry information regarding the ownership or an image which represents the ownership. The copyright images are usually known as stamps which visually modify the underlying image or the document while not totally destroying the visibility of the underlying image or the document. High bit rate watermarks are usually stamps which are embedded invisibly into the images.

In order to embed a binary watermark image into the original image, we need to select a suitable watermark and then place it's information in the image so that it does not corrupt the image, and is detectable only with the knowledge of the encryption keys and/or the original image.

Most natural images provides some capacity to embed additional information without causing noticeable perceptual differences. The amount of such information which can be embedded depends on the image. We expect such embedded information to survive legitimate image processing activities such as compression.

After recent success of image compression standards such as JPEG, we believe it is worthwhile to design the watermarking algorithms specific to the compression standards such as JPEG. We expect this will give us more capacity for information embedding thus suits for high bit rate watermarking.

We target the future JPEG200 standard which is based on wavelet transform. Due to the lack of information of final JPEG2000 compression algorithms, we choose SPIHT [1] as the compression algorithm due to its superior performance and simplicity. We expect the ideas implemented in this paper can be easily modified for JPEG2000. We have designed the embedding scheme to survive significant compression ratio under SPIHT compression, in particular we used the bit-plane ordering of SPIHT and embedded binary watermark variables onto bit-planes.

Proceedings of the Fourth Australasian Document Computing Symposium, Coffs Harbour, Australia, December 3, 1999.

2 Binary Feature Based Embedding Process

The watermark image is represented as

$$W = \{w(i,j) \in \{0,1\}, 0 \leq i \leq 2^{k_1}, 0 \leq j \leq 2^{k_2}\}$$

and image to be watermarked as

$$X = \{x(i,j), 0 \leq i \leq 2^{l_1}, 0 \leq j \leq 2^{l_2}\}.$$

We used grey scale images with 8-bit pixels thus intensity ranging from 0 to 255.

Wavelet transform enables images to be transformed into multiresolution images which enables embedded progressive image coding [1] . In line with this property of the wavelet transform, we transform the watermark image to a multiresolution representation, $\psi_W(W)$, so that we can embed the watermark in a progressive manner. This multiresolution representation of the binary images is discussed in section 6.

For improved security we can encrypt the original watermark image and use this new encrypted image for watermark embedding. A simple technique which can be used is random permutation of watermark pixels. Since we apply a multiresolution transform on to the watermark image, we encrypt the subbands separately so that we do not lose the gains of the multiresolution transform. We call this new binary image as the encrypted watermark image which is represented by $\psi^*_W(W)$.

Any watermark embedding scheme must alter a selected portion or a set of pixels of the original image. We can allocate only small number of bits of the original image to each binary watermark image pixel. In high bit rate watermark embedding we cannot embed a particular image bit to the statistics such as adding a pseudo-random sequence of the given set of pixels due to its smaller size and statistics does not make sense when the size is small. Thus we look for binary features of X so that we can embed the watermark bits by altering these binary features. The particular feature required is that if the binary value is flipped, ie $1 \rightarrow 0, 0 \rightarrow 1$, then there is no perceptual distortion of the image. This feature extraction process is denoted by $F_e(X)$ which take the original image and returns the extracted feature image $f_e(i,j) = F_e(\psi_X(X))$. The way these features are found is explained in section 5.

Now the embedding process is to mix $F_e(\psi_X(X))$ and $\psi^*_W(W)$ so that with the knowledge of $F_e(\psi_X(X))$ or $\psi^*_W(W)$ or nothing we are able to detect the encrypted watermarked image. We will denote the embedding process as $E(F_e(\psi_X(X)), \psi^*_W(W))$ which takes the extracted feature image and the encrypted watermark image and returns the feature image to be updated, $f_u(i,j)$. Details of this embed operator

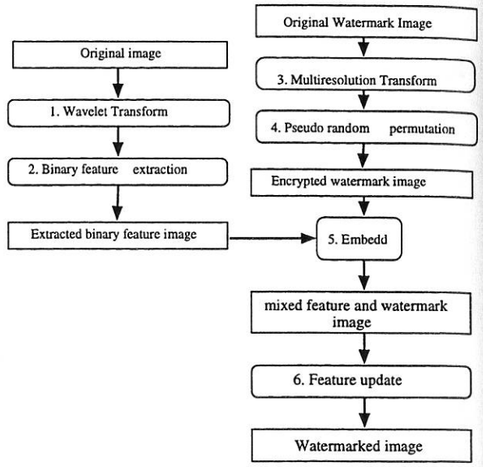


Figure 1: The high bit rate watermark embedding process

is discussed in section 4. Using this mixed feature image we will construct the watermarked image, $x_w(i,j)$. We denote feature update process as $F_u(X, E(F_e(\psi_X(X)), \psi^*_W(W)))$, which takes the original image and the mixed feature image and returns the watermarked image. This feature based high bit rate watermark embedding process is described in Fig. 1.

3 The Detection Process

The detection process is comparatively simpler and is depicted in Fig. 2. Using the same parameters of feature extraction of embedding process, the feature image must be detected from the suspected image. This extracted feature image must be submitted to the detection operator which we will discuss in the next section. The detected feature image will be transformed to suspected watermarked image using the encryption parameters or the permutation matrix used in the embedding process.

This binary image which is in multiresolution representation, is inverse transformed back into the original image using the synthesis binary filters discussed in the section 6. This suspected watermark image will be compared with the original watermark image manually or automatically using a similarity measure.

4 Embed and Detect functions

Now we look for arguments which will reveal properties on Embed(E) and Detect(D) functions. We assume they are bitwise operators. The possible inputs to the detector are encrypted watermark image $p(i,j)$ or extracted feature image $f_e(i,j)$ or nothing from the original image as the possible key

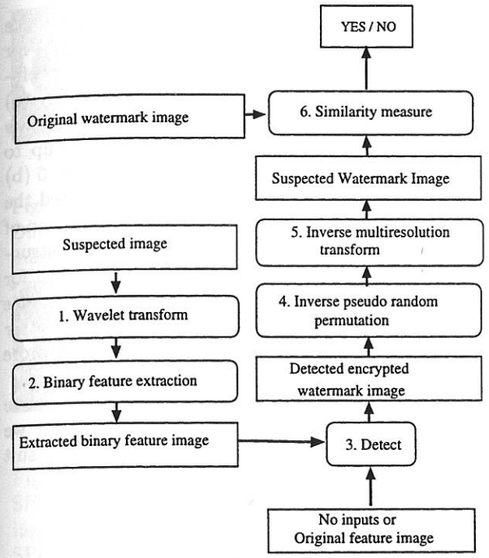


Figure 2: The high bit rate watermark detection process

information together with extracted feature image of the suspected image. If watermarked image is fetched into the detector we get the following two equations,

$$f_u(i,j) = f_e(i,j) \;\; E \;\; p(i,j)$$

$$p(i,j) = (\text{either } p(i,j) \text{ or } f_e(i,j) \text{ or nothing }) \;\; D \;\; f_u(i,j)$$

From the equation only $f_e(i,j)$ makes sense and hence we define the detection operator as,

$$p(i,j) = f_e(i,j) \;\; D \;\; f_u(i,j).$$

The following theorem explains possible choices for the detector and embedder.

Theorem 1 *The only consistent choices for the operators E and D are given by the following table.*

	E	D
1	XOR	XOR
2	$\bar{p}$	$\bar{p}$
3	p	p
4	$NOT\ XOR$	$NOT\ XOR$

Proof: These cases can be proven using the following table.

f_e	p						1	2	3	4
0	0	0	X	1	X		0	1	0	1
0	1	0	X	1	X		1	0	1	0
1	0	X	1	X	0		1	1	0	0
1	1	X	1	X	0		0	0	1	1
Validity		N	N	N	N		Y	Y	Y	Y

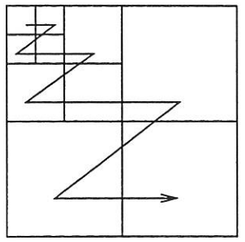


Figure 3: Scan order of coefficients

Only last four columns are the possibilities for identifying the E and D functions. The columns must be consistent in the sense of functions, i.e one-to-one mapping. We can identify these consistent columns by looking to identify f_u from f_e and p, and p from f_u and either f_e or p.

The second and third possibilities correspond to the detector with no inputs except the extracted feature image from the suspected image. Our next question is which is the best choice in terms of protection it provides. If we change the feature bits randomly, irrespective of which embedding operator we use, its going to effect the encrypted watermark image. Thus our criteria for the best choice is simplicity, and protection must be achieved from other parts of the embedding process such as feature extraction and encryption of watermark signal. Also we expect the watermark embedding process is image dependent otherwise an attacker will find it easy to identify the watermark if we embed the same watermark in different images. The choice 1 or 4 is suitable for cases where the feature extraction parameters are image independent as in [2]. Choice 2 or 3 is suitable when feature extraction parameters are image dependent as in our paper.

5 Bit Plane Embedding

Due to compression and perceptual significance all coefficients are not suitable for watermark embedding. We need to embed the watermark bits progressively such that embedded information is tolarent to compression. Our embedding algorithm is designed for bit-plane oriented image coding, in particular for SPIHT image coding algorithms. We embed the watermark bit to the wavelet coefficient by simply making the i^{th} bit plane value equal to the watermark bit while guaranteeing the error within a given bound (distortion step). The higher significant bit planes are coded earlier and hence we use a simple sorting algorithm to identify the coefficients and the corresponding bit planes in the most significant order.

Algorithm 1

- *Scan the subband in the order depicted in figure 3. Denote these coefficients* $c_0, ..., c_N$.

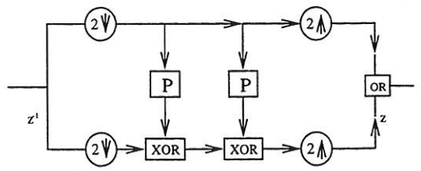


Figure 4: 2-channel Binary Signal Filter Bank

- *Let* $c_{max} = max\{c_0, ..., c_N\}$ *and* $p = 0$.
- *For* $i = \lfloor log_2 c_{max} \rfloor$ *to* $i = \lceil log_2 q \rceil$
  For $j = p$ *to* N
  If $c_j + q$ *flips* i^{th} *bit plane*
  Swap c_p *and* c_j.
  Increment p.

6 Multiscale Transform of the Binary watermark Image

We have chosen the coefficients such that the chosen bitplane values for the watermark bits survive compression. In order to make use of this property we transform the watermark image to a multiresolution representation using the filter bank as shown in figure 4. Successive application of the filter bank to low pass subband yield the multiresolution representation. We used separable filters. Other such binary wavelet filters [3] can also be used.

The filter bank follows the lifting approach [12]. In the analysis side, original signal is separated into even and odd components by the down-sampling operators. The odd values are predicted from the even values and the prediction error is calculated using the XOR operator.

Again, in the synthesis side, odd values are predicted from the even values and original odd values are recovered from the XOR operation. The resulting odd and even values are interleaved using the up-sampling operator to construct the original signal. Notice, the perfect reconstruction of the filter bank is guaranteed since,

$$(o \ XOR \ e) \ XOR \ e) = (o \ XOR \ (e \ XOR \ e))$$

$$= o \ XOR \ false = o$$

where o is an odd value and e is a predicted odd value. Multiresolution representation of our watermark image is shown in 5 (b).

7 Results and Discussion

We have used the lena image in figure 9 and bike image in figure 10 used in JPEG2000 standardisation process for watermarking purposes. The watermarked lena image is given in figure 11 and the watermarked bike image in figure 12 for the distortion step 8 and decomposition levels 5. We have measured the compression performance of the watermarking algorithm under SPIHT compression. We have achieved subjective detection performance up to 9.4:1 compression ratio for lena and 11:1 for the bike image without zeroing of wavelet coefficients as given in figure 6 (a) and in figure 7 (a) respectively. With zeroing of wavelet coefficients we have achieved higher compression ratio up to 23:1 for lena and 32:1 for the bike as in figure 6 (b) and in figure 7 (b) respectively. We also tested the subjective performance with partial construction of the detected watermark. The partial reconstruction of the watermark at compression ratio 32:1 for lena is given in figure 8. The better performance of the bike image is due to its relatively better smoothness which results in the survival of more least significant bits in the compression process.

What is evident in our results is that to achieve watermark survival at high compression ratios we need either to design compression algorithm to suit the watermarking algorithm or vice versa.

References

[1] Said Amir and Pearlman William. A new fast and efficient image codec based on set partitioning in hierarchical trees. *IEEE Trans. on Circuits and Systems for Video Tech.*, Volume 6, pages 243–250, June 1996.

[2] Hsu Chiou-Ting and Wu Ja-Ling. Hidden digital watermarks in images. *IEEE Trans. on Image Processing*, Volume 8, Number 1, pages 58–68, January 1999.

[3] Swanson M. D. and Tewfik A. H. A binary wavelet decomposition of binary images. *IEEE Trans. on Image Processing*, Volume 5, Number 12, December 1996.

[4] Swanson Mitchell D., Zhu Bin and Tewfik Ahmed H. Robust data hiding for images. In *IEEE Digital Signal Processing Workshop (DSP 96)*, Loen, Norway, September 1996.

[5] Kundur Deepa and Hatzinakos Dimitrios. A robust digital image watermarking method using wavelet-based fusion. In *Proc. IEEE Int. Conf. on Image Processing*, Santa Barbara, California, October 1997.

[6] Voyatzis G., Nikolaidis N. and Pitas I. Digital watermarking: An overview. In *Proc. of EUSIPCO'98*, Rodes, Greece, September 1998.

[7] Low S. H., Maxemchuk N. F., Brassil J. T. and O'Gorman L. Document marking and identification using both line and word shifting. In *Infocom'95*, Boston, Massachusetts, April 1995.

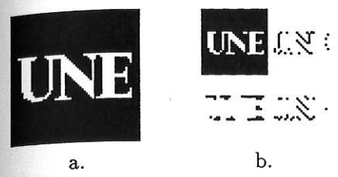
[8] Pitas Ioannis. A method for watermark casting on digital images. *IEEE Trans. on Circuits and Systems for Video Tech.*, Volume 8, Number 6, pages 775–780, October 1998.

[9] Cox Ingemar J., Kilian Joe, Leighton Tom and Shamoon Talal. Secure spread spectrum watermarking for multimedia. Technical Report 95-10, NEC Research Institute, Princeton,NJ, 1995.

[10] Smith Joshua R. and Comiskey Barrett O. Modulation and information hiding in images. In *Proceedings of the First Information Hiding Workshop*, Isaac Newton Institute, Cambridge, U.K, may 1996.

[11] Wolfgang raymond B., Podilchuk Christine I. and Delp Edward J. Perceptual watermarks for digital images and video. In *Proceedings of IEEE*, July 1999.

[12] Sweldens W. The lifting scheme: A custom design construction of biorthogonal wavelets. *Appl. Comput. Harmon. Anal.*, Volume 3, Number 2, 1996.



a. b.

Figure 5: (a) The original watermark image (b) One level multiresolution transform of the watermark image

a. b.

Figure 6: (a) The detected watermark image after SPIHT compression ratio of 9.4:1 with no smoothing. (b) The detected watermark image after SPIHT compression ratio of 23:1 with smoothing for lena image.

a. b.

Figure 7: (a) The detected watermark image after SPIHT compression ratio of 11:1 with no smoothing. (b) The detected watermark image after SPIHT compression ratio of 32:1 with smoothing for bike image.

Figure 8: The detected watermark image with partial reconstruction after SPIHT compression ratio of 32:1 with smoothing for lena image.

Figure 9: The original lena image

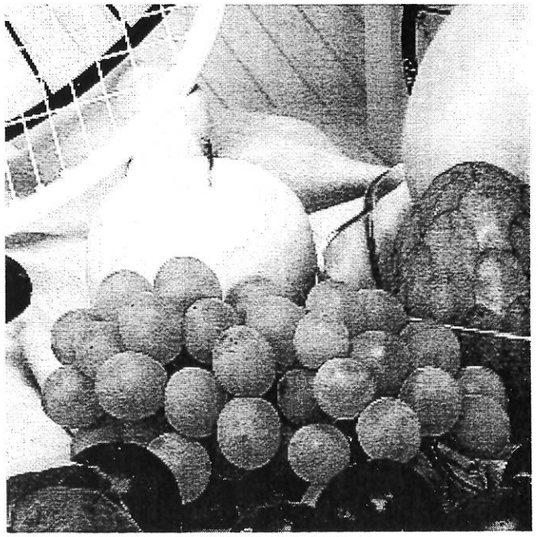


Figure 10: The original bike image

Figure 11: The watermarked lena image

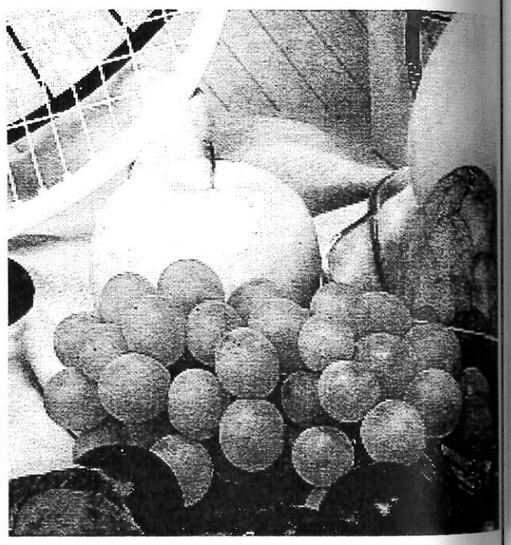


Figure 12: The watermarked bike image

The use of argumentation to assist in the generation of legal documents

John Yearwood

School of Information Technology and Mathematical Sciences

University of Ballarat, Ballarat, Victoria, Australia

j.yearwood@ballarat.edu.au

Andrew Stranieri

Donald Berman Laboratory for Information Technology and Law, Dept of Computer Science and Computer Engineering

La Trobe University, Bundoora, Victoria, Australia

stranier@cs.latrobe.edu.au

Chaula Anjaria

School of Information Technology and Mathematical Sciences

University of Ballarat, Ballarat, Victoria, Australia

c.anjara@ballarat.edu.au

Abstract

Many text documents in the legal domain are created in order to express the reasoning steps a decision maker followed in reaching conclusions. For example, refugee law determinations are documents that express the reasoning steps a member of the Refugee Review Tribunal in Australia followed in order to infer conclusions regarding the status of an applicant. Although, it is reasonable to expect that a mapping between the reasoning steps used by a decision maker and the structure of the document produced would clearly be apparent, a number of authors have discovered that such a mapping is by no means obvious. In order to develop legal knowledge based systems that generate documents from their own reasoning steps, discourse analysis is invoked to bridge the gap and perform the mapping. In this paper, we articulate a heuristic that we use to generate a plausible document structure without the use of discourse analysis. Without discourse analysis, the heuristic cannot contribute to our understanding of the process employed by decision makers to convert reasoning to text. Nevertheless, the heuristic can mimic the process. The heuristic has been trialed with a small sample of refugee law determinations by extracting the reasoning steps from each determination and applying the heuristic to reproduce each document's structure.

Keywords Document generation, argumentation, refugee law.

1 Introduction

In many applications of human reasoning, conclusions ultimately reached and the reasoning steps employed to reach conclusions are expressed in written natural language. For example, the inferences that members of the Refugee Review Tribunal (RRT) make in assessing claims for asylum seekers to remain in Australia as refugees are natural language documents called determinations that vary from 6 pages to many tens of pages in length and only loosely conform to a pre-defined structure. However, although determinations reflect reasoning, each one is written in a style that is not obviously consistent with a representation of refugee reasoning that we have developed over the last two years in close collaboration with RRT members.

The disparity between a natural language document and a representation of reasoning that a document expresses has been noticed by a number of authors using different knowledge representation schemes so is likely to be an artifact of communication styles rather than a peculiarity of any one knowledge representation scheme. Dick [2], in translating legal decisions that spanned hundreds of years into conceptual graph frames initially attempted to do so by creating graphs directly from text components. This proved to be too difficult because of the disparity between the document text and a conceptual graph representation of the reasoning that the document expressed. She proceeded by reading each judgement in its entirety, then formulated a set of conceptual graphs from her understanding and not from the text.

Proceedings of the 4th Australasian Document Computing Symposium,
Coffs Harbour, Australia,
December 3, 1999.

More recently, Branting *et al* [1] reiterate the absence of isomorphism between an inference tree that represents domain reasoning and a rhetorical tree that represents the organisation of text in a document and go further, first, by clearly articulating the benefits of a mapping and then by utilizing discourse knowledge to bridge the gap. According to those authors, the discourse structure is particularly clear in documents such as contracts or wills and therefore well suited to natural language generation techniques aimed at planning and realising multi-sentential text generation.

Multi-sentential text generation has application in tasks such as contracts or wills because these tasks re-use similar documents in a manner that becomes difficult for a human to perform quickly and in large numbers. For these applications an automated system that generates as much of the finished document, as quickly as possible, is preferred. However, automated document generation systems can be useful to support decision makers in those tasks that involve less re-use of similar documents and more individual crafting of text to suit the case at hand. The drafting of a refugee determination is unlikely to ever be automated like a wills generator can be, because the life and death nature of a determination necessitates, for political if not for moral reasons, the full involvement of human decision makers. Nevertheless, a document drafting system can support, without crossing the imaginary line towards replacing, the decision maker by generating a document structure and leaving the creation of sentences to the author.

We agree with Branting et al [1] that the discovery of a mapping between reasoning steps and document structure based on discourse knowledge leads to improved document generation. Indeed, it seems unlikely that a complete mapping between reasoning steps and document structure can ignore discourse analysis. However, a rigorous mapping is not necessarily required if the sole objective is to convert a series of reasoning steps into a plausible document structure. In this study we propose a relatively simple heuristic for the traversal of an inference tree that realises a structure that is similar to that created by authors. The heuristic cannot explain why a particular document structure derives from an inference tree. Some invocation of discourse analysis would be required for that level of description. Nevertheless, in the restricted domain of refugee law determinations, the heuristic can be applied to generate a document structure that is similar to one that a human author would plausibly arrive at. The heuristic can therefore be used as the basis of an automated document generation system that is much simpler than one based on discourse knowledge.

There is quite a large variation between what are considered to be discourse segments and discourse relations in the field of discourse analysis. In the Rhetorical Structure Theory of Mann and Thompson [6] there are the notions of non-overlapping text spans called *nuclei* and *satellites* with rhetorical relations such as *elaboration* which holds between them. In the Grosz and Sidner Theory (GST)[5], the elementary text units are called discourse segments and the discourse is explicitly stated to be a tree. Each discourse segment is characterised by a primary intention called the *discourse segment purpose.* GST identifies only two kinds of intention-based relations between discourse segments: dominance and satisfaction precedence. A large amount of research in discourse planning is based on the speech act theory proposed by Searle [8]. Dietz and Widdershoven [3] identify limitations of Searle's theory put forward by Habermas [4] and go further by noting that communication support systems based on Habermas's theory are quite different from those based on Speech Act Theory. The mappings between rhetorical relations and speech acts are problematical as are the mappings between intentional and informational levels. Marcu [7] provides a melding of text structures and intentions by formalising a discourse structure with nodes characterised by four features: the status of the node, the rhetorical relation that holds between the nodes that are immediate children, the set of salient units and the primary intention. We propose investigating this structure with the cases studied because there have been many instances where the explanation for a discourse segment relates to underlying intentions of the member that augment the reasoning.

This work achieves a transformation from a reasoning structure to a draft document structure that fits with the examples studied. Without the use of sophisticated discourse planning the user of a decision support system is able to refine a document which presents a relatively complete record of their reasoning. They may then be involved in a refinement process that consists of removing points that they would prefer to have implicit, leaving the salient points and possibly elaborating on these.

In this paper, we initially outline the knowledge representation schema we use in order to illustrate that it is relatively complex and therefore can capture intricate associations. Following that, we discuss our experience in extracting reasoning steps from a determination. Our objective is to demonstrate the effectiveness of the heuristic by reproducing the structure of the determinations in our sample. Results indicate that a relatively simple heuristic can mimic the structure of determinations and therefore be used as the basis of a system that supports the decision maker by generating a document outline that displays a plausible structure.

2 Argument based knowledge representation

The knowledge representation we used is a variation from the Toulmin argument structure [10]. Despite the immediate appeal of the Toulmin argument structure (TAS) as a convenient frame for representing knowledge, most researchers that use Toulmin structures vary the original structure. A survey of the different variations of the Toulmin structure can be found in [9]. The statement "Most Saudis are Muslim" in Toulmin's now famous example, may be a warrant that convinces us that the assertion "X is a Muslim" follows from the knowledge that "X is a Saudi". However, this warrant communicates two distinct meanings. On the one hand the warrant indicates a reason for the relevance of the fact "X is a Saudi". On the other hand the warrant can be interpreted as a rule which, when applied to the fact that "X is a Saudi" leads us to infer that "X is a Muslim". These two apparent meanings are best perceived as different roles the warrant has in the structure of an argument. Drawing the distinction between the two roles a warrant has in an argument, leads us to explicitly identify four features that are left implicit in the Toulmin formulation. Figure 1 illustrates Toulmin's Saudi argument using our representation. The four features made explicit are:

- reasons that explain why a data item is relevant for a claim. The warrant in the original structure is seen as a reason for the relevance of the data item. In other examples that Toulmin uses, his warrant statement is a rule that we equate to an inference procedure.
- an inference procedure, algorithm or method used to infer a claim value from datum variable values. This must be made explicit if a computational model is to be built because conceivably any computable function can be used to infer a claim value from data item values.
- a {variable : value } formulation of each claim and data. This formulation is convenient for the application of an inference procedure.
- reasons that explain why the inference method used is appropriate. In some examples the inference procedure is a neural network. Reasons for the appropriateness of the inference procedure relate to how well the network was trained. In many legal arguments, a statute may make explicit an inference rule, in which case, the statute is the reason for appropriateness of the inference procedure.

We call the argument in Figure 1, a generic argument. An actual argument is an instance of a generic argument in that that values for variables are set. For example, an actual argument may infer X is almost certainly a Muslim from data that indicates X is certainly a Saudi. The Reasons for relevance of a data item and reasons for the appropriateness of a inference procedure apply to all actual arguments instantiated from a generic one regardless of variable values. However, a component we call a claim value reason is sometimes necessary in order to encode a reason for why a particular claim value follows from a particular data item value.

We have identified over 200 generic arguments that have been used by members of the Refugee Review Tribunal or applicants for refugee status by perusing determinations and interviewing members. Figure 2 illustrates the generic argument that makes the claim regarding well founded fear of persecution. No inference procedure has been specified for this argument as research is initially in progress toward the use of the knowledge representation to support human inferences and not to implement machine inferences.

The arguments combine to form a chain or tree of arguments because one argument's claim is another argument's data item. For example, another argument exists that has, as its claim, The applicant (does, does not have) a real chance of persecution , which is one of the data items in Figure 2. The ultimate argument asserts an applicant (is, is not) a refugee. In the next section we shall describe the representation of a sample determination using our schema and contrast this with the text of the determination.

3 Juxtaposition of reasoning steps with document structure

The first sample refugee determination involves a Sri Lankan woman who was determined to be in need of refugee status by the Refugee Review Tribunal. Her claims for refugee status were based on an incident some years ago in which her home was burnt down and more recent incidents of beatings and other reprisals by the armed forces following contact with persons suspected of belonging to the rebel army, LTTE. The determination is 5 pages in length and contains the following headings: *Jurisdictional Foundations* (Two paragraphs common to all determinations that establish the authority of the Tribunal to review the case); *The Law.* (Eleven paragraphs common to all determinations that outline leading cases, statutes and other relevant background information); *Findings of Fact.* (This section contains the member's summary of all claims an applicant has made. Associated with a summary of each claim is the member's own finding regarding the veracity of the claim and accompanying reasons.); *The Decision* (Two paragraph statement of the decision).

Figure 3 represents a partial tree of claims extracted from this determination. Reasons for the relevance of data items, appropriateness of inference procedures or for particular claim values, are omitted in Figure 3 for brevity. The claim labelled A with value (is) indicates the applicant is a refugee. This is made on the basis of the value of one data variable, B. B is the claim for an argument in its own right and has data items C and D. Claims in Figure 3 stop at N for brevity. In total there were 36 claims identified in the determination.

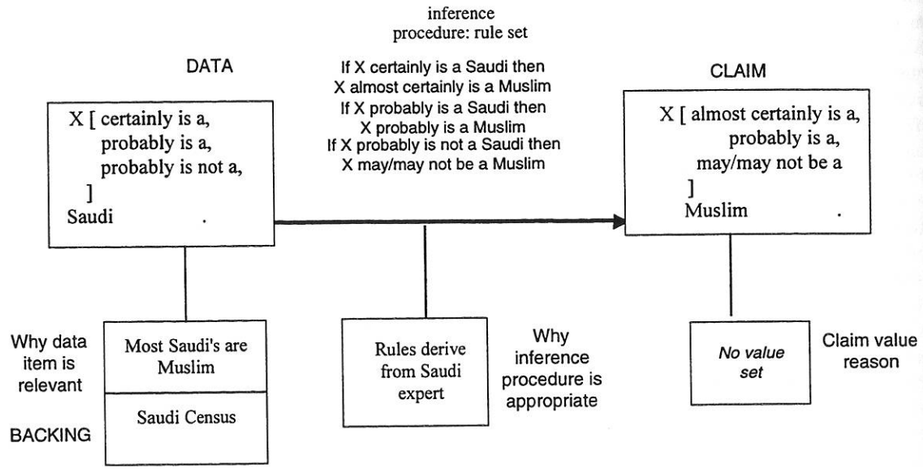


Figure 1. Our variation to the Toulmin Structure

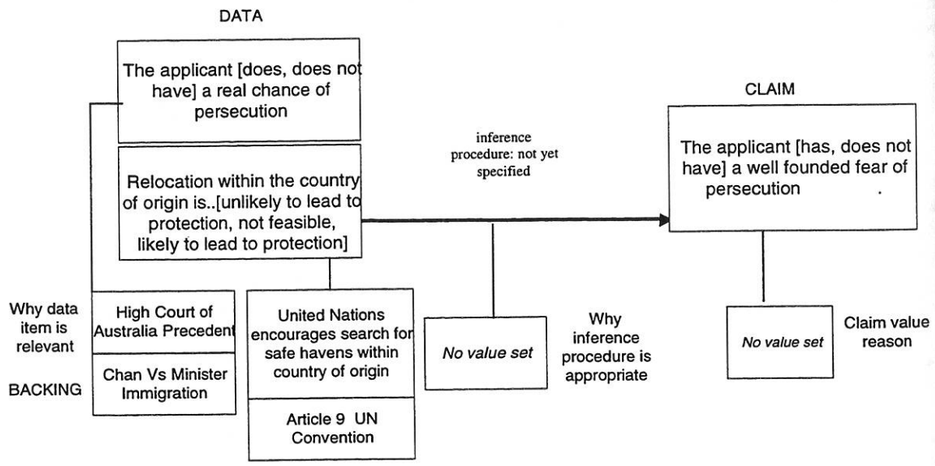


Figure 2: Generic *Well-founded fear* argument

The values for claims were extracted manually by reading the case, tagging passages relevant for a claim and assigning the value that seemed best to capture the member's intended meaning. For some claims the appropriate passage and value was obvious. For example, claim A, the variable *refugee* is unequivocally taken to have the value (*is*) from our reading of the following paragraph in the determination:

The application for a domestic protection (temporary) entry permit is remitted for reconsideration with a direction that the criterion requiring the applicant to be a non-citizen who has been determined to be a refugee under the Convention and Protocol, is satisfied. (1)

The variable, G, *Taken together all incidents constitute harassment* is associated with the text fragment:

The burning of the applicant's home and belongings in Trincomalee, was a deliberate act designed to punish the applicant and her daughter for their perceived connection with the LTTE. Such an act does not carry the imprimatur of international approval as a reasonable use of force for purposes of self protection. Furthermore, I am satisfied that the applicant's daughter was actively persecuted thereafter by Sri Lankan military authorities in a totally unacceptable manner. I find that both the applicant and her daughter, XX, were persecuted by the Sri Lankan authorities prior to their departure. (2)

There could be found no text fragment that directly corresponded to some variables in our representation. For example, variable J describes the extent to which an incident impacted on the applicant in a severe or persecutory manner. We surmise that the member has inferred that the house burning incident in Case 1 did impact severely on the applicant because of the tone of the text associated with variable G, depicted here in the paragraph above. We surmise that the member has left text fragments associated with this, and many other variables, implicit.

The reasoning expressed in the determination was found to map onto 36 variable:value tuples contained within 10 arguments. Passages of text such as the examples above mapped directly onto 18 of the 36 variables. Values for the remaining 18 variables were not stated explicitly in the determination, but like variable J, were left implicit. A value for the *well founded fear* variable is similarly not explicitly mentioned in the text but is assumed by the context of the entire document to have the value *has*

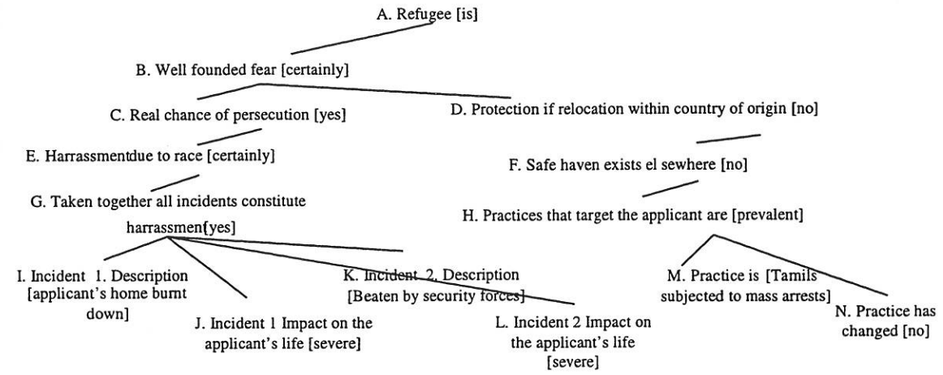


Figure 3. Partial tree representing reasoning steps in case 1

Traversal	Sequence
Top down, breadth first traversal	A, B, C, D, E, F, G, H, I, J, K, L, M, N
Bottom up	I,J,K,L,G,E,C, M,N,H,E, F,D, A
Actual	1..16, I,K,F,M,G,E,C,38,39, A.
Bottom up, sub-tree constrained traversal	I, {J}, K, {L} F, M,{N},{H},G,E,C,{B}{D}A

Table 1. Sequence of passages associated with reasoning steps of Example 1

Twenty cases were randomly selected for analysis. Text fragments from three of the cases were associated with variables from our representation independently by three assessors. The majority of text fragments were associated with the same variable:value pair by each assessor. However, there were some anomalies. Each anomaly originated from some ambiguity concerning the discourse intention of the member. For example, a fragment in a case involving a Timorese applicant was associated with

the *well founded fear* variable by two assessors and with the *credibility* variable by the other assessor. The disagreement was due to subtleties in interpretation of the intention underpinning the text. This is a clear indication that rigorous analysis of text requires discourse analysis representation and that speakers intentions need to be taken into account. However, the anomalies occurred sufficiently infrequently to suggest that, in the majority of refugee determinations, the member's intention is clear. This obviates the need for sophisticated discourse analysis if the objective is solely to support the member in generating a base determination for further refinement.

Table 1 illustrates two possible traversals of the tree in Figure 3; a top down, breadth first and a bottom up variant. The same table also indicates the sequence within the determination that the passages actually appeared and a traversal we call bottom up, sub-tree constrained that we describe below.

Paragraphs 1..16 and 38..39 do not correspond to any of the generic arguments we used to model refugee determinations. Paragraphs 1..15 comprise the standard paragraphs in the *Jurisdictional Foundation* and *The Law* sections that provide background information but do not relate specifically to the reasoning steps in the case at hand. Paragraphs 16, 38 and 39 relate to the age of the applicant and although a generic argument does not exist we can represent the reasoning made in these paragraphs with the inclusion of an additional and new argument.

As expected, a top down or bottom up traversal of the inference tree does not obviously map neatly onto the actual structure of the document. However, a traversal that we call *bottom up, sub-tree constrained* represents a heuristic traversal that does map the document structure neatly onto the inference tree. This traversal is described informally as follows:

1. Begin at any node near or on a leaf
2. Visit all descendent nodes, that is, all nodes within the sub-tree defined by the node, in any order until there are no nodes unvisited in the sub-tree
3. Where possible, insert implicit nodes adjacent to siblings
4. Select any unvisited node and repeat.

The traversal I, {J}, K, {L} F, M,{N},{H},G, E, C,{B}{D}A is consistent with the heuristic as follows:

- Begin at node I because it is a leaf,
- I has no sub-tree
- Insert implicit node {J}
- Visit any node not visited: K
- K has no sub-tree, Insert implicit node {L}
- Visit any node not visited, F
- Visit any node in the sub-tree of F: M
- M has no sub-tree, Insert implicit node {N}
- Visit any node in the sub-tree of F: {H}
- Sub-tree F completed so visit any unvisited node: G
- All nodes in sub-tree of G have been visited so visit any unvisited node: E
- All nodes in sub-tree of E have been visited so visit any unvisited node: C
- All nodes in sub-tree of C have been visited so visit any unvisited node: {B}
- Visit any node in the sub-tree of {B}: {D}
- All nodes in sub-tree of {B} have been visited so visit any unvisited node: A

The traversal heuristics do not produce a document structure isomorphic to the inference tree. This is not a limitation because the expression, in text, of the same reasoning by different persons is not expected to be identical. The objective of a traversal algorithm is to produce any good document structure and not the one optimal document however optimal is defined.

The traversal heuristics have intuitive appeal because Members, in weaving a determination, do not commence at the conclusion because there are too many issues that have contributed to a value at that level. Instead an issue closer to the facts (leaves) are selected and all facts and issues relevant to that issue are discussed (ie. all sub-tree nodes). Once that issue is covered, the author has creative freedom to select any other issue. Discussion pertaining to that issue continues until all issues and facts relevant to that are covered. While it is certainly possible that an author exercising a skillful degree of flair may construct a document using heuristics other than these, our contention is that the heuristic presented here can be used to generate a document that appears at least, coherent to a reader, even if it lacks dramatic emphasis.

The algorithm that describes this traversal relies on a global data structure that contains the list of nodes that have been visited in a tree. This is initialized to null before the algorithm, traverse, is called for the first time, with the root of the tree as argument.

The function pickANode selects any node in the tree with root, R that has not already been visited and is near the leaf nodes. The function nodesToVisit terminates the traversal of the tree/sub-tree when there are no nodes unvisited in the sub-tree.


```
traverse(N)

Purpose: visit nodes in a tree using bottom up, sub-tree constrained traversal

Arguments: R, node label that is the root if the tree to be traversed

Updates: variable visitedNodes

Calls: nodesToVisit, pickANode, traverse

While nodesToVisit (visitedNodes, R) = true
   Do N gets pickANode(visitedNodes, R)
      If N <> null
         Then begin
                  Append N to visitedNodes
                  Traverse (N)
                 end
      endif
endwhile
end traverse.
```

In order to illustrate our traversal we take a second example, drawn at random from a pool of determinations. In the second case, the applicant is an ethnic Tamil citizen of Sri Lanka from Jaffna with a heart condition who experiences some incidents of harassment whilst in Colombo. Return to his home in Jaffna would involve a real chance that his life would be endangered because of a government embargo on medical supplies to the area (directed at LTTE sympathizers). Relocation to the most likely area of Colombo is not possible because none of the threshold criteria for returnees to establish themselves without serious security problems can be met.

Table 2 illustrates that the bottom up, sub-tree constrained traversal described by our heuristics reproduces the actual argument if the split passages associated with, variable N are united. Although quite different factually, examples 1 and 2 both contained 10 Toulmin arguments and over thirty variable:value attributes. In both examples there was a high proportion of implicit variables (18/36 for example 1 and 18/31 for example 2). In summary, example 2 confirms the observations we made from the first example.

To date, the reasoning steps in twenty cases drawn at random from a pool of RRT determinations have been extracted using the knowledge representation frame described above. The heuristic we described has been applied to each of the twenty cases in order to ascertain whether the structure of the actual document could be produced using our heuristic. The heuristic was found to account for the actual structure of the document in each of the twenty cases. We plan to survey member's opinions of document structures generated with the use of the heuristic. Table 3 illustrates a case that is one of the more intricate we sampled. In this case the reasoning steps involved 58 variable:value tuples. Twenty eight were explicitly represented by text fragments and 30 were implicit. The sequence of fragments once again conforms to the heuristic described above.

Traversal	**Sequence**
Actual	N,K,N,9-16,N,L,M,C,25,F,I,O,P,J,D,A
Bottom up, sub-tree constrained traversal	N, K, L, M, {G}, {H}, C, F, I, O, P, {Q}, J, D, {B}, A

Table 2. Sequence of passages associated with reasoning steps of Example 2

Traversal	**Sequence**
Actual	P1-11,22,24,25,Y,27,26,1,29,15,3,B,5,32,7,35,F,9,21,F,C,H,F,P36-P39, C, F, W, X, L, F, C, B, D, P51, A
Bottom up, sub-tree constrained traversal	22, {23}, 24, 25, {11}, {V}, L, W, X, {M} {I}, Y, 26, 27, {28}, {12}, {13}, {2}, {N}, 1, 29, 30, {31}, {16}, {15}, {2}, {O}, 3, {4}, {P}, 5, 32, 33, {34}, {6}, {Q}, 7, 35, {36}, {37}, {18}, {19}, {8}, {R}, 9, {20}, 21, 10, S, {J}, {E}, F, C, {T}, {V}, {K}, {G}, H, D, B, A

Table 3. Sequence of passages associated with reasoning steps of a third case

4 Conclusion

In performing a fine-grained analysis of the mapping between a document structure and the reasoning steps expressed in a document we noticed that a mapping is not obvious. The heuristic that we use to generate a plausible document structure without the use of discourse analysis has proved to be successful in the tests so far. Without discourse analysis the heuristic

cannot contribute to our understanding of the process employed by decision-makers to convert reasoning to text.

Nevertheless, the heuristic can mimic the process sufficiently well to be useful in the task of supporting document drafting in a complex domain where there will always be a need for human interaction and refinement. Future research in this direction is aimed at formalising the heuristics and engaging in more rigorous evaluations. An evaluation should include comparisons of documents generated with our heuristics with actual documents in addition to studies that measure the readability or coherence of documents generated with our heuristics.

5. Acknowledgements

This research was supported by the Refugee Review Tribunal, Australia and the Australian Research Council.

6. References

[1] Branting, L., K., Callaway, C., B., Mott, B., W., and Lester, J., C., 1999. Integrating Discourse and Domain Knowledge for Document Drafting. *Proceedings of Seventh International Conference on Artificial Intelligence and Law ICAIL'99.* ACM Press. pp214-220.

[2] Dick, J. P. 1991. *A conceptual, case-relation representation of text for intelligent retrieval.* Ph.D Thesis. University of Toronto. 1991. Canada.

[3] Dietz, J. L. G., and Widdershoven, G. A. M., 1992. A comparison of the linguistic theories of Searle and Habermas as a basis for communication supporting systems in van de Richt, R. P. and Meersman, R. A. (Eds)., 1992. *Linguistic Instruments in Knowledge Engineering.* Elsevier Science Publications. Pp121-130.

[4] Habermas, J., 1987. The theory of communicative action. (tr Thomas McCarthy). Boston : Beacon Press

[5] Grosz, B. J, and Sidner, C. L. 1986. Attention, Intentions and the structure of discourse. *Journal of Computational Linguistics.* Vol 12, No 3. Pp175-204.

[6] Mann, W, C. and Thompson, S. A., 1988. Rhetorical Structure Theory: Toward a functional theory of text organisation. *Text* Vol 8, No. 3. pp 243-281.

[7] Marcu D. 1997. The Rhetorical Parsing, Summarisation, and Generation of Natural Language Texts. Ph D Thesis, Department of Computer Science, University of Toronto.

[8] Searle, J., 1969. *Speech Acts: An Essay in the philosophy of language.* Cambridge University Press. Cambridge.

[9] Stranieri, A., and Zeleznikow, J. 1999. A survey of argumentation structures for intelligent decision support. *Information Systems and Decision Support Systems ISDSS'99* Melbourne July 1999 Monash University Press.

[10] Toulmin, S. 1958. *The Uses of Arguments.* Cambridge University Press. Cambridge

Automatic document metadata extraction and manipulation: a working system for the Intelligence Analyst

Mark Burnett

DSTO
Dept of Defence
Fern Hill Park, Bruce, ACT

mark.burnett@dsto.defence.gov.au

Richard Jones

Lloyd-Jones Consulting
PO Box 6155
Philip ACT 2605

jonesrl@compuserve.com

Abstract

This paper discusses the design and implementation of an operational system to aid health intelligence analysts. The HINTS system provides automated support to undertake tasks such as specific health-related research and report writing in the face on an ever-growing body of electronic information, available on the web, and on local file systems. Our approach is to provide automated support for document analysis and discovery from technologies that support ad-hoc searching, consistent filtering for specific pieces of information such as hospital facilities, diseases and locations, and that provide document summarisation and keywording. Document metadata is stored in XML in a data structure that allows a variety of searches and views of the document space to be performed. The user interfaces to the system by web browser and a map-based geospatial application.

Keywords
Document Analysis, Document Databases
Information Retrieval, XML,
Information Extraction

1 Introduction

Intelligence analysts typically operate in two modes. In the first mode, they scan, on a regular basis, a wide range of documents from a range of sources, in case they contain something that might be useful. A selection is made on some general criteria and those documents are put to one side for a rainy day, organised in some way so that they can be found later. They move into the second mode of working when called upon to prepare a specific brief. They obtain the source material for the brief by accessing this repository of information, and combining this with the results of specific searches from other sources. Invariably the information that is key is a side issue in a document that may be discussing some other topic. This information extraction requirement is, of course, the focus of the MUC series of experiments [1].

This paper describes a repository system to support both modes of operation described above, but with more emphasis on the first. The system, known as HINTS, is based upon an XML store containing three types of information, standard bibliographic information, domain specific information - in this case the health domain, and information specific to the intelligence analysis requirement. A key part of the system is to assign specific values to metadata elements automatically. To this end an information extraction process looks for specific entities (e.g. disease names, locations), and a document summariser assigns keywords and extracts a summary.

A strong goal of the design was to make the system generic and easily applicable to other intelligence domains. To this end we selected technologies that operate on text at a surface level, and a component-based architecture for integration.

2 XML Data Storage

We chose an RDF implementation of XML to provide a generic storage mechanism, and to allow concepts, and relations among concepts, to be defined. RDF also supports a class system analogous to Obect-Oriented programming and modeling systems. The RDF specification has now progressed to a "Proposed Recommendation" to the W3C (see http://www.w3c.org/TR/PR-rdf-syntax).

3 Metadata Extraction

It is often useful for users to read some sort of condensed or structured surrogate for a document. The purpose may be varied: it may be to determine if the document is worth reading in full, or to extract specialised information from the document, either by reading specific portions, or even without reading it at all. Some standard structures to support these requirements, (e.g. MARC) and abstracts, keywords, etc. have been in place for a long time. This

information is designed to tell anyone, regardless of their role or background, what a document is about.

Bibliographic metadata does not assist if the user requirement is to find a particular fact, such as an instance of an environmental event, like a volcanic eruption, or references to particular entities such as a company or person. This information may be imbedded in a document whose main topic is something completely different, and it is impractical to extract such objects without some detailed knowledge of the interest of the reader. Such information can certainly be seen as metadata, being domain specific or even user specific.

The existence of metadata also provides a set of indexing mechanisms to enable particular documents to be retrieved, with improved precision over full text retrieval. e.g. Find Jones as an author.

A major problem for determining content-based metadata has been the effort and skill required to create or extract it. Over the past 10 years, and especially in the past five years, techniques have been developed to try to automate some of these processes with greater or lesser success. The US government has given a major impetus to such techniques through the TREC and MUC conferences.

Two software packages were used to provide the automatic assist in metadata generation in HINTS:

- FXBench, and
- InTEXT Analyser

3.1 FXBench

FXBench was developed in DSTO in the Electronic and Surveillance Research Laboratory in Salisbury, South Australia. It comprises a suite of tools that can assist a Language Engineer to identify patterns in text, and to write Fact Extractors that produce formatted data from documents.

Fact Extractors are of two forms:

- Closed vocabulary lists of nomenclature (e.g. disease lists) where words or phrases can be mapped onto a preferred form; and
- Pattern definitions defining regular expressions using the PERL syntax.

HINTS uses FXBench to fill in domain specific metadata fields.

3.2 InTEXT Analyser

InTEXT Analyser was developed by InTEXT Systems Inc. It is based upon AIDA, the summarising and keywording software developed by Computer Power Group in conjunction with the Australian Federal Parliamentary Information Systems Office in the early 1990s (Jones [2] and Thistlewaite [3]). Another variant of this software was used in TREC 4 (Burnett [4]) in an experiment to determine the effectiveness of establishing a reduced full text index using only the automatically extracted keyphrases.

HINTS uses InTEXT Analyser to extract keyphrases, and sentences to fill in the Keyword and Summary metadata fields.

4 Technology Integration

The architecture used to integrate the various technological components employed an n-tier distributed set of components. It is a complex architecture incorporating:

- a web server (MS Internet Information Server);
- an XML data server (eXcelon)
- DCOM and CORBA components; and
- a Client/Server (JavaBeans) Component.

The HINTS operational prototype runs on an NT server, with browser and Java-application clients. The user interface is described in the next section.

5 Searching And Viewing The Metadata

Dynamic views and searches of the data in the metadata repository are served out to web clients and to a Geospatial Java application.

5.1 Browser access

A browser provides access to the following functional features:

i. Metadata Basic Search;
ii. Metadata view by Entity;
iii. Metadata Advanced Search;
iv. Modify Metadata;
v. Basic Search ;
vi. Advanced Search;
vii. Content Classification;

Features v-vii are provided by UltraSeek, a commercial text retrieval product available from InfoSeek. It was configured to index both the internal HINTS collection of documents, and a set of external health-related web sites.

Searching, viewing and modifying the metadata repository used active server pages, linked with server-side extensions of the eXcelon server, to access the XML store.

The screen shot shown in Figure 1 shows part of a document hitlist following a metadata query.

Viewing by location, medical facility, hazardous animal, or disease is available for the current set of documents from a drop-down menu.

These dynamic views allow greater access to the information stored in the repository, and show how different entities and concepts are related across a set of documents.

Document Info	Health Intelligence	Abstract	Key Phrases
Title: unknown THAILENV.DOC 22-02-1995	DISEASES LOCATIONS Thailand [10] Bangkok [4] Cambodia [1] Malaysia [1] Burma [1] Laos [1] MEDICAL FACILITIES ANIMAL HAZARDS Snake [2] Spider [1] Centipedes [1] Scorpion [1] Mosquito [1]	THAILAND Information Cut-off Date: February 1994.0 ENVIRONMENTAL HEALTH RISKS OF OPERATIONAL IMPORTANCE ENVIRONMENTAL HEALTH RISK ASSESSMENT: Safe food storage and sanitary food handling practices vary greatly, but are virtually nonexistent among str ... see more	water, Korat Plateau, ENVIRONMENTAL HEALTH RISK, significant health hazards, Thailand, populated, western mountains, northern, sq km, Widespread deforestation,
Title: unknown SGENV.DOC 12-08-1998	DISEASES LOCATIONS Dakar [6] Frederick [2] Fort [2] Sand [1] Bower [1] MEDICAL FACILITIES ANIMAL HAZARDS Snake [2] Scorpion [1] Spider [1] Centipedes [1]	ENVIRONMENTAL HEALTH COMPONENT OF THE DISEASE AND ENVIRONMENTAL ALERT REPORTS (DEAR) SENEGAL Information Cutoff Date: January 1995.0 The environmental health component of the DEAR, prepared by the Armed Forces Medical Intelligence Center (AFMIC), pro ... see more	ENVIRONMENTAL HEALTH RISK ASSESSMENT, Senegal, water, environmental health component, Mean Daily Maximum/Minimum, average annual rainfall, region, urban, sq km, sq mi,

Figure 1: Document metadata.

The screen shot in Figure 2 is a location view of the document set following an "ebola" query.

The locations contained in the document set are presented as an ordered list, with the most commonly occurring location at the top. For each location, the set of documents that contains that location is shown as a sub-list. These documents are ordered by their relevance to that term, with a colour-code indicating the value of the relevance.

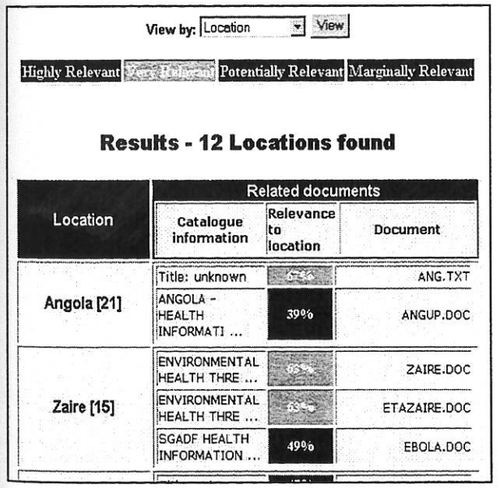


Figure 2: A location view.

5.2 Geospatial access

A geospatial interface is a natural interface to documents that often deal with one or more locations. Using a highly detailed gazetteer, location names are mapped to co-ordinates that allow data to be displayed on a map of the world. Using a health layer in the OpenMap(tm) package (http://javamap.bbn.com:4711) a user can display document information on a map at various magnifications, and drill down to very detailed views of a region.

Figure 3 is a screenshot showing the results of an "ebola" query, with folders indicating documents containing occurrences of ebola.

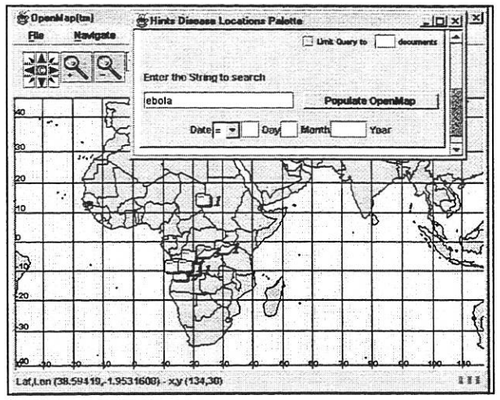


Figure 3: Querying the metadata via Openmap.

6 Conclusion

This paper has sketched the design and implementation of a generic system for intelligence analysis. Particular attention has been paid to the extraction and use of document metadata.

References

[1] *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, Morgan Kaufmann, San Francisco, 1998.

[2] R.L. Jones. AIDA - the Artificially Intelligent Document Analyser, *In Libraries and Expert Systems, ed. McDonald and Weckert,* Taylor Graham, pp 49-56, London, 1991.

[3] P.B.Thistlewaite and S.Blume. Offloading Information Overload: the AIDA Project. In *Proceedings of the Seventh Conference for Librarians in the Criminal Justice System*, Canberra, January 1990.

[4] S.M Burnett, C. Fisher and R.L. Jones - InTEXT Precision indexing in *TREC4. ed. : D. K. Harman NIST Special Publication 500-236,* pp 287-294, 1995. http://trec.nist.gov/pubs/trec4/t4_proceedings.html

Describing and Viewing Large User Models

James Uther
Department of Computer Science
University of Sydney
Sydney, Australia
jimu@cs.usyd.edu.au

Judy Kay
Department of Computer Science
University of Sydney
Sydney, Australia
judy@cs.usyd.edu.au

Abstract

We are developing ways to allow medical students access to user models built on data obtained by online quizzes. A students' individual model can be viewed in a Java applet on a web browser. To provide the model to the the applet, and perhaps to enable exchange of models, we have developed a simple schema for the user model in the Resource Description Format (RDF). The schema and visualisation are described here.

Keywords RDF, User Models, Visualisation

1 The Model

The Graduate Medical Program (GMP) at the University of Sydney uses a problem based curriculum. The lack of formal course outlines in such programs often induces students to over-study. To overcome this the GMP offers an 'online assessment' system in which students can try questions relevant to the course from a bank of around five thousand questions supplied by the course writers [5]. These questions mark the standard of learning required. The students' progress is tracked by the system for feedback to the student, and to question writers.

The model is broken into *learning topics*, the atomic units of the curriculum. In the GMP a learning topic is a unit of study small enough to be summarised with references on two pages. There are approximately 570 learning topics in the first two years of the four year course. Each of these topics has a location (a URL), a title, and is related to other topics by keyword or some other categorisation. Each topic can also have a number of scores associated, calculated from the data collected by the online assessment system. In this trial we included the average mark for the modelled user for each topic, the average mark for each topic of the users' year, and a mark for each topic that the curriculum authors think that the modelled student should be reaching at the time the model is generated.

Proceedings of the Fourth Australasian Document Computing Symposium, Coffs Harbour, Australia, December 3, 1999.

1.1 RDF Representation

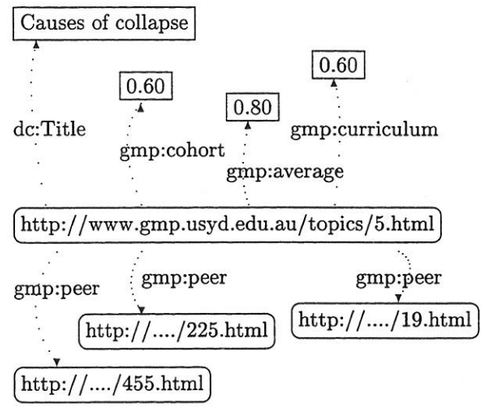


Figure 1: RDF Graph of a model entry.

An RDF [7] graph of a node of the model is shown in Figure 1. The central resource is a learning topic named by its URL. This resource has a title (Causes of collapse), some marks, and some related resources (known as peers in this schema). These peers may then have their own titles, marks, and further peers.

```
<rdf:Description about=
  'http://www.gmp.usyd.edu.au/topics/5.html'>
<dc:Title>Causes of collapse</dc:Title>
<gmp:peer rdf:resource=
  'http://www.gmp.usyd.edu.au/topics/455.html'/>
<gmp:peer rdf:resource=
  'http://www.gmp.usyd.edu.au/topics/225.html'/>
<gmp:peer rdf:resource=
  'http://www.gmp.usyd.edu.au/topics/19.html'/>
<gmp:curriculum>0.60</gmp:curriculum>
<gmp:average>1.0</gmp:average>
<gmp:cohort>1.0</gmp:cohort>
</rdf:Description>
```

Figure 2: A fragment of the XML Serialisation of the RDF Graph in Figure 1, ignoring namespace declarations.

We serialise the graph for the node to XML as in Figure 2. The full model includes relevant XML namespace declarations which are not shown here. Where possible we reuse well known schemas, in particular the Dublin Core [1] for the Title property. The peers are written as multiple properties of the same name. The alternative was to use the *rdf:Bag* construct, but it was felt this would complicate parsing of the file on the client. The scores in the model are named from an internal schema. We are looking at using a more accepted schema that covers user performance like the IMS metadata set [2].

2 Visualisation

The model is described in the RDF schema shown above, but must still be shown to the student in an easily digested form. The visualisation of the model must

- allow the person viewing the model to get a good understanding of the overall state easily.
- allow the person viewing to find detailed information about specific topics.
- allow the student to set a level of performance they are happy with, rather than impose a fixed 'pass' mark.
- allow comparisons between models. For instance a student should be able to find out which on topics they are ahead or behind their year.
- show the relationships between topics.
- allow the viewer to run on a variety of platforms, possibly over modem connections, and on clients with reasonable processors (200Mhz) and XGA (1024x768) screen resolution.
- leave enough space on the screen to display a full web page.

The model viewer is a Java applet placed in one frame of a web page, beside a larger pane that can display a learning topic page, or online assessment questions relating to a learning topic. This leaves us with perhaps 300x550 pixels in which to display the model of 600 items each with additional information and relationships.

Our visualisation can be seen in Figure 3. We show the graph of topics as a vertical list of items reminiscent of DEXTER [9]. The graph is shown as a spanning tree from the selected topic, with depth in the tree represented through brightness, font size, and spacing of the topic titles. The current data set (user average, year average, a comparison, etc) is shown by the hue of the topic titles. Green

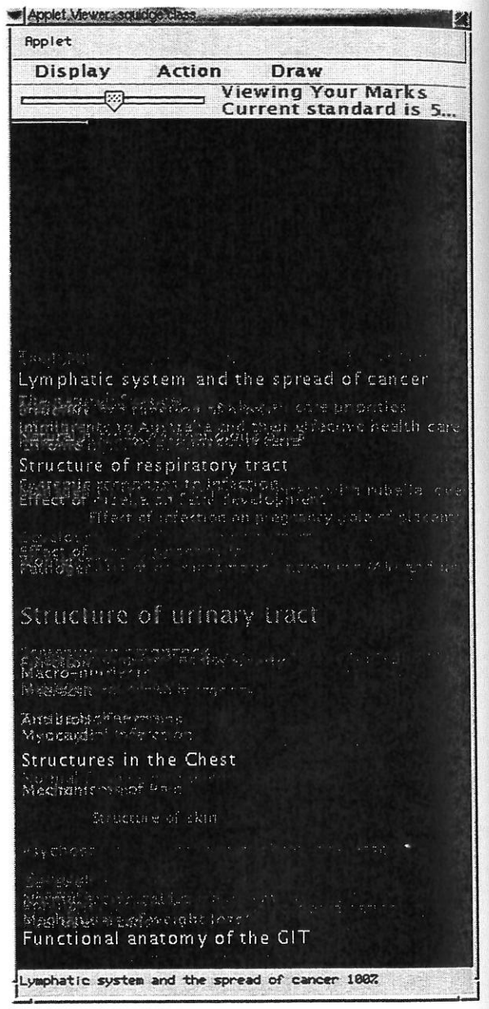


Figure 3: The user model viewer

indicates a higher value than a standard, red a lower, and yellow no information. Degree is shown by colour saturation.

The user interacts with the display by clicking on topic titles, which selects them, calculates a new spanning tree, and updates the display accordingly with an animation. The user may view more detailed information about the selected topic by selecting options from a menu at the top of the viewer, which then requests the appropriate web page and displays it in an adjacent frame of the web browser. Currently the options available are

- view the learning topic itself
- do online assessment questions from that learning topic
- see the evidence for the model on that learning topic

Other menus on the viewer select which data set to show (user average, year average and so forth), and allow some optional interface tweaks to be turned on or off as desired. There is an option to allow the user to drag topics around, thus uncluttering cluttered areas of the display. However, this is initially disabled to avoid confusing new users. Another option (activated in Figure 3) shifts to the right those topics for which there is no current information. This makes it easier to select topics for which there is some information, these being generally more interesting.

2.1 Implementation Experiences

The model RDF file is generated by a small Java program from the online assessment results stored in a RDBMS. This takes several seconds on a lightly loaded machine, making generation on request impractical. Further work on the model storage infrastructure and some optimizations should allow much faster generation.

The client parses the RDF file using a standard XML parser and the SAX parser API. The parser generates objects representing each topic resource. A routine then generates some indexes on the list of topic objects, and *weaves* the topics together by generating references from topic objects to their peers. This weaving process, necessary to be able to find a spanning tree in reasonable time, tends to be the slowest step on average hardware, so a progress indicator is shown.

The visualisation itself is handed a reference to the graph of topics, and spends a small amount of time generating further indexes, mapping topics to screen positions and the like. The animation uses a simple iterative algorithm. First the final position of all topics after the move is calculated. The topics are then all moved half the distance from their current position to the final position and redisplayed. This step repeats six times. Finally all topics are drawn in their final position. The space left around a topic is calculated by taking the inverse of the square of the depth in the tree of the topic, so most space is given to the topic at the top of the tree, and exponentially less space to topics further down the tree.

The implementation has so far proved fast enough to manage the display and animation of seven hundred topics on a modern Java virtual machine and current hardware.

3 Benefits and Drawbacks of RDF

RDF is simply a method of describing internet resources by attaching attributes to a resource by a labeled arc. The resulting directed labelled graph can then be serialised to XML. Most importantly, this XML file can then be restored to exactly the same graph using the published formalism, enabling transfer of a fairly generic representation of the metadata between applications. We have so far only written the visualisation client, but any client that has some knowledge of what our arcs and nodes are (the schema) could consume our file and reason with it [8, 6], or display it differently. There is a proposed way of specifying RDF schemas [4], which we intend to use to formalise our schema when the proposal is accepted as a standard.

Our RDF graph was structured rather simply to avoid difficulty in parsing and building data structures on the client. Recently a number of libraries for handling RDF structures have appeared [10, 11, 3] that try to define generally useful operations on an RDF graph in the same way that the DOM defines generally useful operations on an XML tree. This implementation predates these projects and so has not used them, although we watch the projects with interest. It is also doubtful that these generalised graph structures would be fast enough in their current implementations for our specific purposes, although current work on 'triple databases' within the Mozilla project [3] is interesting.

4 Acknowledgements

Much of this work was supported by the Department of Educational Development and Evaluation, Faculty of Medicine, University of Sydney.

References

[1] Dublin core metadata initiative, http://purl.org/DC.

[2] The ims project, http://www.imsproject.org/.

[3] The mozilla project, http://www.mozilla.org/.

[4] Dan Brickley and R.V. Guha. Resource description framework (rdf) schema specification, http://www.w3.org/TR/PR-rdf-schema/.

[5] Simon Carlile, Stewart Barnet, Ann Sefton, and James Uther. Medical problem based learning supported by intranet technology: a natural student centred approach. *International Journal of Medical Informatics*, 50:225–233, 1998.

[6] Stefan Decker, Dan Brickley, Janne Saarela, and Jürgen Angele. A query and inference service for rdf. In *QL'98 - The Query Languages Workshop*, 1998.

[7] Ora Lassila and Ralph Swick. Resource description framework (rdf) model and syntax specification, http://www.w3.org/TR/REC-rdf-syntax/.

[8] Massimo Marchiori and Janne Saarela. Query + metadata + logic = metalog. In *QL'98 - The Query Languages Workshop*, 1998.

[9] Michael Murtaugh. The automatist storytelling system. Masters thesis, MIT, MIT Media Lab, 1996, http://ic.www.media.mit.edu/icPeople.hide-/murtaugh/thesis/index.html.

[10] Eric Prud'hommeaux. Perl rdf parser and triple database, http://www.w3.org/1999/02/26-modules/announce.txt.

[11] Janne Saarela. Sirpac - simple rdf parser & compiler, http://www.w3.org/RDF-/Implementations/SiRPAC/.

Intranet Search Using Content and Metadata
(Demonstration)

David Hawking*
CSIRO Mathematics and Information Sciences,
Canberra, Australia
Peter Bailey and Nick Craswell
Department of Computer Science, ANU
Canberra, Australia
David.Hawking@cmis.csiro.au

November 6, 1999

Abstract

A search engine has been developed in the ACSys Cooperative Research Centre for use in Intranet environments in which web pages tagged with search metadata co-exist with un-tagged pages. It provides a mechanism for combining metadata constraints with full-text queries in a sensible way. The search engine has been adopted for use within the Australian National University and within CSIRO.

The query language and interface will be demonstrated and a brief description given of the architecture and key algorithms. The ACSys search engine is capable of indexing data very quickly (over 2 million web pages per hour on an $8,000 PC) and of fast query processing. The techniques used have achieved top-level results in international evaluation. Efficiency and effectiveness results will be presented.

`http://search.anu.edu.au/`

*The authors wish to acknowledge that this work was carried out within the Cooperative Research Centre for Advanced Computational Systems established under the Australian Government's Cooperative Research Centres Program.