ADCS'OO Keynote Address

Web Wombat - Building the Premier Australian Search Engine

Phillip Betolus
Technical Director
Web Wombat
www.webwombat.com

December 1,2000

Abstract

The Web Wombat story from a technical perspective (excluding the nitty gritty secret stuff). Starting at the origin of the Web Wombat search engine concept and the initial engines, developed in the back room of the house, through to search engines running from various parts of the world. There are also some forward looking statements about the future of search, which may or may not come to pass and do not reflect any particular business plans Web Wombat may or may not have.

About the Speaker

Phill Bertolus is Technical Director and cofounder of Web Wombat. Ph ill has 20 years experience in computing environments as diverse as aviation, banking, military. and Internet industries and has written about telecommunications and mobile phones for Australia's leading newspapers (The Age. The Sydney Morning Herald and the Australian Financial Review).

He wrote a compiler by the age of 23 and has successfully developed and internationally marketed a number of PC products.

His special areas of interest include compiler design, search engine design and realtime computing. Phill is a computer engineering specialist with a BE (Electronic Engineering) and a DipE (Communications Engineering).

Towards an Efficient Retrieval of Medical Imaging

Richard CHBEIR *Franck FAVETTA*

Laboratoire d'Ingénierie des Systèmes d'Information (LISI)
20 av Einstein, F-69621 Villeurbanne - France
Phone: (+33) 4 72 43 88 99 - Fax: (+33) 4 72 43 87 19

richard.chbeir@ieee.org *franck.favetta@ieee.org*

Abstract

Image description is not an easy task. The same image can be described through different views: on the basis of either low-level properties, such as texture or color; context, such as date of acquisition or author: or semantic content, such as real-world objects and relations. Our approach consists in providing a global description solution capable of integrating different dimensions (or views) of a medical image. Via our approach, we are able to propose a solution that takes into consideration the heterogeneity of user competence (physician, researcher, student, etc.) and a high expressive power for medical imaging description. Visual solutions are recommended and are the most suited for non "novice" users in computing. However, current visual languages suffer from several problems as imprecision and no respect of integrity of spatial relations. Particularly, resolution of ambiguities generated by the user and/or the system at different levels of image description remains a challenge. In this paper, we present our solution for resolving these issues. A prototype has been implemented.

Keywords: Information Retrieval, Medical Imaging, Spatial Relations, Ambiguity Resolving.

1 Introduction

Image description is person-dependent and confined to parameters such as context, image objects, application domain, etc. The current approaches take into consideration only certain image facets. In a historical museum, image retrieval is based on the semantic content linked to objects and the scene (context) relating them. In monitoring applications, the searching process is built upon dates (all images at 13/12/1999). shapes. colour and texture.

We have found that current approaches suffer from different weaknesses:

1- Incomplete description of image content: an interpretation of a satellite image of the moon may ignore other stars.
2- Ambiguities resulting from the incompatibility between system pertinence and user relevance.
3- Impossibility of reutilization: a database of medical x-rays in hospital "A", for instance, cannot be reused in hospital "B" where physicians exercise another specialty because acquisition and retrieval systems are domain-dependent.

A recent study has begun in our laboratory **(LISI)** [9, 10, 11] aiming to provide an adequate image retrieval solution for heterogeneous users in the medical domain. We propose a solution built on **global description** process at storage process which takes into account complexity and variety of medical imaging (MRI, Scanner, X-rays, etc.). If image description during storage process takes into account more elements and different points of view, retrieval process becomes efficient for global and various types of queries. In this paper, we present in particular how ambiguities formulated by the user during query process can be resolved.

The next section details related work in this domain. Section 3 presents our data model. Section 4 presents the description process used by our system. Section 5 shows ambiguities problems and solutions. The last section concludes and summarizes our future directions.

2 Related work

In the literature, the description problem of images has been accessed through different approaches:

- **The contextual approach** considers only external data such as acquisition date, author name, file name or artificial keys [23]. However, this is restrictive and inappropriate for several domains such as Medical Imaging, TV production, multimedia, art history, geology, satellite image databases, etc.
- **The abstraction approach** manipulates the physical (low-level) properties of an image such as color histogram [4], texture [6, 12], shape [20. 7], edge [19, 21], etc. The abstraction approach is applied to domains where various types of images exist such as TV databases, museum databases, etc. This approach is currently having great interests because of automatic procedures used to compare images. However, abstraction approach is not appropriate in medical domain because of the diversity of the human body and the time-consuming procedures required for each digitalization techniques (Scanner, **MRI,** X-ray, etc.).
- **The classification approach** treats high-level properties of the image and describes its semantic content in terms of real objects and relationships. Object description and classification are generally done manually (or semi-automatically) at image storage. Describing the content and the sense of each object is difficult because probable descriptions are numerous and each person may describe the image differently. Secondly, image description based on object position and relationships between objects (spatial facet) has proven to be imperfect at retrieval process where translation, scaling, perfect and multiple rotations, or any arbitrary combination of transformations is applied. Thirdly, a great waste of data is induced when replacing an image by a set of poor semantic descriptors to describe image objects.

As the main objective of our project consists in providing retrieval medical imaging destined for non-professional end-users, user-friendly languages are required. Several language types have been proposed the last decades [2, 16, 24]. Current retrieval systems proposed in the medical domain [1, 13, 25] are not user-oriented and only few query possibilities are proposed. Visual languages can be seen as precursors in many domains applications. They aim at supplying user-friendly interfaces, especially for handling of spatial criteria in spatial queries. Several approaches have been proposed [5, 8, 17, 22], presenting advantages and drawbacks in terms of ambiguity and user-friendliness [18]. The main advantage of visual languages comes from the fact that the user does not have any constraint to express a query and no new language to learn. Nevertheless, many limitations still remain. The main limitation conies from the ambiguities of visual languages.

3 Data model

Image content is very rich in terms of properties, characteristics. salient objects and relationships. Fundamentally, the medical image content represents an aggregation of **salient internal** image objects: *organs*, *atomic regions*, and *anomalies*. Each salient object possesses low-level features (texture, shape, color, etc.) and high-level features (semantic noun, definition, synonyms, etc.). Based upon Eakins framework [14], our model integrates four dimensions:

- The **contextual dimension** regroups global and external image properties (external object) without taking image's content into account.
- Each salient objects can be mapped into three orthogonal dimensions:
  - The **physical dimension** regroups local physical properties of image content as colors, textures and other low-level features.
  - The **spatial dimension** takes into consideration shapes (polygons) and spatial relations (cardinality and topology) calculated between objects.
  - The **semantic dimension** integrates semantic objects and relations.

4 Description process

To support our proposition, an implementation called MIMS (Medical Image Management System) has been realized [9, 10. 11]. The description of the whole system is out of scope of this paper. The architecture of MIMS is surveyed here below, followed by a short exposition of the topological precision of MIMS, and a description of the *ambiguity resolver* component.

4.1 Architecture

MIMS is composed of several components (Figure 1): analyzer, ambiguity resolver, SQL Translator, and Spatial Knowledge Model (SKM). Each component has its own task. In short, the analyzer assists the user to describe the image. The ambiguity resolver checks ambiguities found in each dimension (semantic, spatial, and physical). The SQL translator transforms commands into SQL statements. The Spatial Knowledge Model, based on Spatial Knowledge Base and the Inference Base, plays the consultant task and guides the ambiguity resolver to decide, whenever an ambiguity is found, whether the system or the user is able to respond. The analyzer, in collaboration with the Spatial Knowledge Base (a component of SKM), is responsible for providing possible features to describe image dimensions (spatial, physical,

Proceedings of the 5th Australasian Document Computing Symposium, Twin Waters Resort, Sunshine Coast, Australia, December 1, 2000.

semantic, and contextual). In essence, the spatial knowledge base (SKB) is a set of structures (spatial, physical, hierarchical, semantic and evolutive) that embed different medical image dimensions.

MIMS analyzer passes by several steps:

- Contextual analysis: this step is used to acknowledge the contextual dimension.
- Physical analysis: this step attempts to describe physical dimension of medical image.
- Spatial analysis: after physical analysis, objects' (and regions') locations and positions can be calculated. MIMS calculates two kind of spatial relations: *directional relations* describing *cardinality* (North, East, Left, Bottom, etc.) and *topological relations.* The next subsection describes the topological expressive power of MIMS.

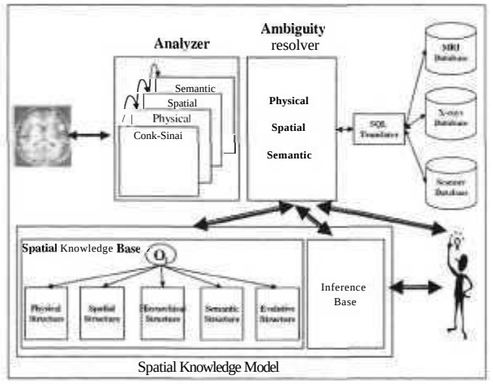


Figure 1: MIMS Architecture.

- Semantic analysis: treats semantic terms (objects and relations) proposed by the user during description process. Spatial Knowledge Model verification is applied during semantic dimension identification.

After semantic analysis, the system sends features to the SQL translator. The latter transforms features translated by ambiguity resolver into SQL statements and sends them to the JDBC[1] driver. In its turn, the JDBC driver examines SQL statements and accesses relative databases (MRI, X-rays, etc.). *Figure 2* shows the analyzer's results of a description process during the storage process.

[1] Java DataBase Connection (JDBC): permits to connect and remote heterogeneous databases using an API supplied in Java Development Kit.

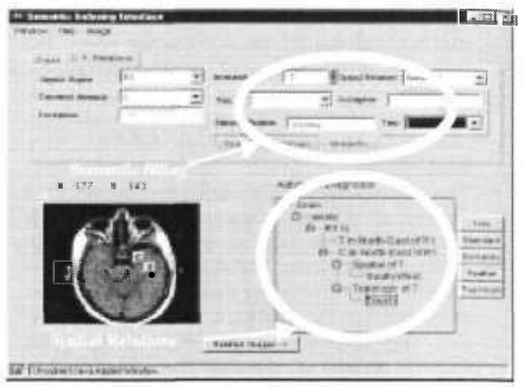


Figure 2: Screen shot of storage interface where image description is realized.

4.2 The topological precision

Topological relations constitute one of the fundamental concepts necessary for the description of spatial data. They are preserved under topological transformations such as translation, scaling, and zooming. The *9-Intersections* model [15] gives a formal categorization of topological relations involving two regions without holes in R_2 *(Figure 3). A* new model, based upon the previous model, gives a formal categorization of all the possible topological relations with three spatial objects in R_2: the *20-Intersections* model (Figure 5).

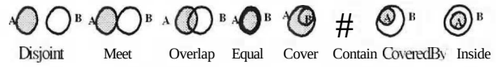


Figure 3: The *9-Intersections* model [15] categorizes eight topological relations between two regions in R_2.

A set of objects with defined *binary* relations can have several configurations which are not *differentiated*, e.g., the two configurations of Figure 5 both have the same *binary* topological description <a1 Meet a2> ^ <a2 Meet a3> ^ <a3 Meet a1> but they are different. To make this distinction is important, especially for the medical domain which requires a high topological expressive power, e.g., the difference between two configurations for the three anomalies of Figure 5 can be crucial for the physician. The *20-Intersections* model provides such a higher topological expressive power.

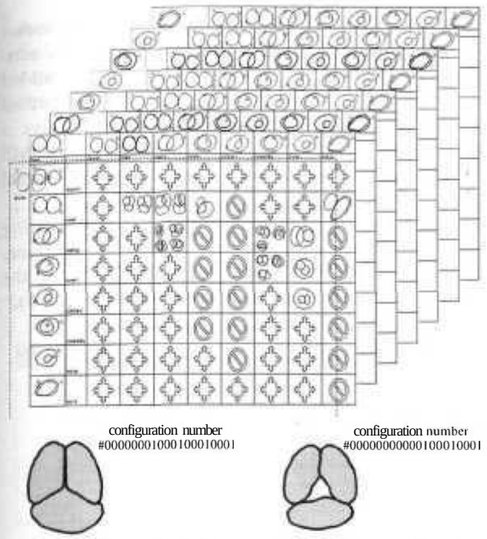


Figure 4: The *20-Intersections* model categorizes 313 possible topological relations with three regions in R_2. Here are two examples of configurations.

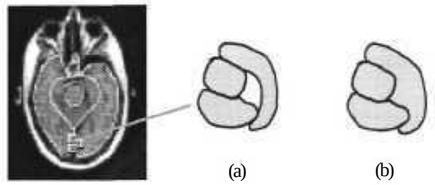


Figure 5: An example of three anomalies in a medical region. In the medical domain, it is crucial that the system can topologically distinguish these two configurations.

Moreover, after modification of spatial relations, it can occur that the integrity of spatial relations can be not respected, e.g.. the configuration <A *Contain* B> ^ <B *Contain* C> ^ <C *Disjoint* A> is impossible. To respect topological integrity, we need to know the spatial expressive power of topological relations with more than two regions. The *20-Intersections* model provides the needed expressive power for three regions.

4.3 Ambiguity resolver

Obviously, certain ambiguities are translated after analysis:

- Physical ambiguity consists in mismatching between semantic objects declared by the user and those of low-level features. MIMS does not currently have such ambiguity because manual physical dimension detection is used.
- Semantic ambiguity occurs when semantic objects or relations proposed by the user are not identified. MIMS currently proposes to differently identify an object or to integrate it into SKB. Whenever semantic relation is defined as a set of spatial ones, ambiguities of semantic relation are resolved at spatial ambiguity level.
- Spatial ambiguities in terms of spatial positioning. As it is known, medical objects location is so complicated to be defined. Next section details how MIMS resolve spatial ambiguities.

The existence of the inference base allows resolving certain ambiguities. The Inference base contains a set of rules used to help in decision-making. This component is upgraded manually but we are working on an automatic upgrade.

5 Spatial ambiguities

The proposed interface in MIMS is hybrid (visual and textual solution), able to give accessibility to a large panel of users. The visual aspect meets important problems of spatial ambiguities. Currently, only few studies have been done to solve ambiguities in visual languages. A taxonomy of ambiguities [18] shows that they are met at different levels and sublevels, making ambiguities one of the most important challenges in visual languages.

A model has been formally defined in [3] for visual systems, which handles ambiguous interpretations of images. This formalism helps us to describe the whole visual environment and to understand where ambiguities can be found. According to the authors, a *visual sentence* is a triple *<i.d,<int,mat»* where *i* is an *image*, *d* is a *description*, *int* is an *interpretation function* and *mat* a *materialization function.*

Figure 6 shows a representation of the formalism. The *image* is expressed by means of a *pictorial language.* The *description* is expressed by means of a *description language.* The functions *int* and *mat* link *images* with *descriptions.* Several *images* can *materialize* the same *description.* This formalism allows to manage multiple visual representations that convey the same meaning, as needed by different users for different tasks.

Figure 6: Formalism of a visual language.

We can use this formalism to describe the whole visual environment (Figure 7). The environment encapsulates the system and the user. These two entities communicate through an interface, including pictorial and textual languages. To allow the user to communicate his actions, the interface provides *inputs*. *Inputs* use metaphors, e.g., the user can click on an icon. *Inputs* can also be geometrical shapes, drawn by the user on the screen. The shape drawn can represent the real shape of a spatial object. In MIMS, to visually add or modify a spatial object, the user can directly act on the medical image. For instance, to add an anomaly, he can click on the medical image at the location of the anomaly. He can choose to *materialize* the object with an icon or a generic anomaly shape supplied by the system. The user can also add an object by drawing an object shape on the image. The system checks if the object shape matches a generic anomaly shape.

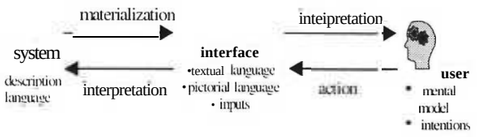


Figure 7: Visual environment.

Figure 7 shows that both user and system must *interpret* the interface. The ambiguity comes from the inteipretation made by the user's viewpoint as well as by the system's viewpoint. Each is capable of making several interpretations. However, which inteipretation is the right one? Ambiguities can be described by the possible answers to the two following questions:

1. How must the system *interpret* user actions?
2. How must the system *materialize* objects and operators to be correctly *interpreted* by the user?

The most important problem with ambiguities is that it induces a lack of precision. It is useless to have a topological model that provides a high topological expressive power if a user description can be interpreted in several ways (several possible configurations in the topological model). For instance, the user *materializes* three anomalies with three icons (Figure 5). He places an icon near the two other icons in the way that all the icons are adjacent. The system can interpret two topological configurations involving the three anomalies.

Several solutions are possible [18]. The first one is that MIMS provides to the user a *clear language,* i.e.. a non-ambiguous textual language. After having visually added or modified spatial objects, the spatial relations derived are displayed in a textual language, in order to give a possibility for the user to check exact semantics. MIMS allows also the user to textually add and modify spatial relations.

Another solution to ambiguities is to establish a *dialog* with the user whenever an ambiguity occurs. For instance, in the previous example, the system shows all the available configurations and requests a choice *(Figure 8).*

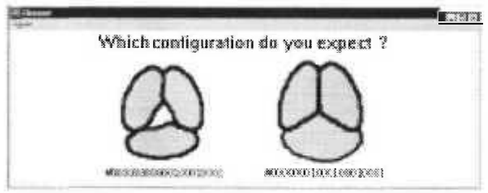


Figure 8: To resolve ambiguities, the system shows all the available configurations and requests a choice.

When checking the respect of the integrity of topological relations, the inference base can find multiple configurations available. The system has to choose one. To solve this ambiguity, the system establishes *dialogs* with the user.

6 Conclusion

The main motivation of this work is to provide a flexible system for answering heterogeneous user queries in medical image databases. Our approach consists in proposing a global image description to achieve an efficient retrieval process able to mix domain and user needs. Ambiguities resolving constitutes a big challenge in medical interfaces where precision is so important in therapeutic treatment and diagnostic analysis. This paper describes our proposition aiming to resolve ambiguities produced by both system and user during storage and query processes. We have implemented a prototype called MIMS (Medical Image Management System). It regroups a set of elements. Spatial Knowledge Model (SKM) is one of the main elements that provides coherent and effective objectivity of image description and analysis. SKM is composed of two components: a Spatial Knowledge Base to assist description and analyzing process, and an Inference Base to check user actions and to manage ambiguities. MIMS is accessible on the WEB and it is currently experimented by several physicians and medicine students. First results and statistics are so satisfactory particularly in terms of high-leveled precision required by the medical domain.

We are currently working on integrating automatic methods able to retrieve low-level features and then to facilitate the user task. Moreover, another study is currently underway which aims to automate SKB structures.

References

[1] Abad-Mota S., Kulikowski C "Semantic Queries on Image Databases: The IMTKAS model". Proc. Of the Basque international Workshop on Information, BIWIT 95, IEEE. Computer Society Press, P. 20-28.

[2] Angelacio, B., Catarci, T.. Santucci, G.. "A Graphical Query Language with Recursion", IEEE Transactions on Software Engineering, Vol 16(10). 1990, P. 1150-1163.

[3] Bottoni, P.. Costabile. M.F. Levialdi, S., Mussio, P.. "Formalising visual languages". In Proc. IEEE Symposium on Visual Languages `95, 1995, P. 45-52.

[4] Ashley J., Flickner M.. Hafner J., Lee D., Niblack W., Petkovic D. "The Query By Image Content System". Proceedings Of the 1995 ACM SIGMOD International Conference on management of data. SIGMOD 95. P. 475.

[5] Aufaure M.-A.. Bonhomme C. LVIS. "A High Level Visual Language for Spatial Data Management", Visual'99. Third International Conference on Visual Information Systems, Amsterdam, The Netherlands, June 2-4, 1999.

[6] Baldi G.. Colombo C. Del Bimbo A.. "Compact and Retrieval-Oriented Video Representation Using Mosaics". Proceedings of Third International Conference, Visual'99, Amsterdam, June 99. P. 171-178.

[7] Bertino E. and Catania B., "A constraint-based approach to shape management in multimedia databases". Multimedia Systems, 1998, P. 2-16.

[8] Bonhomme C, Trépied C. Aufaure M.-A., Laurini R.. "A Visual Language for Querying Spatio-Temporal Databases". ACM GIS99. Kansas City, USA, November 5th-6th, 1999.

[9] Chbeir R., Amghar Y.. A. Flory, "MIMS: A Prototype for medical image retrieval", the 6th international conference of Content Based Multimedia Information Access. RIAO 2000. April 2000, P.846-861.

[10] Chbeir R., Amghar Y., Flory A., "Image Modeling for Medical Databases", ISC A 15th international conference on computers and their applications, CATA 2000, March 2000, P.24-28.

[11] Chbeir R., Amghar Y., Flory A., "System for medical image retrieval: the MIMS model", Third International Conference, Visual'99, Amsterdam, June 99 Proceedings. Page 37-42.

[12] Chetverikov D.. "Pattern regularity as a visual Key", In Proc British Machine Vision Conf., 1998. P. 23-32.

[13] Wesley W. Chu, Fellow, Chih-Cheng Hsu, Alfonso F. Cardenas, and Ricky K. Taira, "Knowledge-Based Image Retrieval with Spatial and Temporal Constructs". IEEE Transactions on Knowledge and Data Engineering, Vol. 10, No. 6, November/December 1998,

[14] Eakins J.P.. "Automatic image content retrieval: are we going anywhere?", In Proc. Of the 3rd International Conference on Electronic Library and Visual Information Research. May 1996.

[15] Egenhofer, M.. Herring, J., "Categorising Binary Topological relationships between Regions, Lines, and Points in Geographic Databases, in: A Framework for the Definition of Topological Relationships and an Algebraic Approach to Spatial Reasoning within this Framework", Technical Report 91-7, National center for Geographic Information and Analysis, University of Maine, Orono, 1991.

[16] Egenhofer. M.. "Spatial SQL: A Query Presentation Language". IEEE Transactions on Knowledge and Data Engineering, Vol. 6 (1), P. 86-94.

[17] Egenhofer. M.. Query Processing in Spatial Query By Sketch. Journal of Visual Language and Computing. Vol 8 (4). 1997. P. 403-424.

[18] Favetta. F., Aufaure M.-A., "About Ambiguities in Visual G1S Query Languages: a Taxonomy and Solutions", International Conference on Visual Information Systems (Visual'2000), Lyon. France. November 2-4, 2000.

[19] Gelgon M. and Bouthemy P., «A Region Tracking Method with Failure Detection for an Interactive Video Indexing Environment », Third International Conference. Visual'99, Amsterdam. June 99 Proceedings. P. 261-268.

[20] Huet B. and Vailaya A.. " Relational Histograms for shape indexing", IEEE ICCV'98, Jan 1998, P. 563-569.

[21] Ikonomakis N., Plataniotis K.N., Venetsanopoulos A.N., "A Region-based Color Image Segmentation Scheme", Proc. Of the SPIE: Visual Communications and Image Processing, vol. 3653, 1999, P. 1202-1209.

[22] Meyer, B., "Beyond Icons: Towards New Metaphors for Visual Query Languages for Spatial Information Systems", Proceedings of the first International Workshop on Interfaces to Database Systems (R. Cooper ed.), Springer-Verlag, 1993, 113-135.

[23] Shakir, Hussain Sabri, "Context-Sensitive Processing of Semantic Queries in an Image Database System", Information Processing & Management, Vol. 32, No. 5, 1996. P. 573-600.

[24] Zloof, M., M.; "Query-By-Example: A Database Language", IBM Systems Journal, Vol 16(4), 1977, P. 324-343.

[25] Zweigenbaum, "MENELAS: An Access System for Medical Records Using Natural Language", Computer Methods and Programs in Biomedicine, 1994, P. 117-120.

Keyword Association Network: A Statistical Multi-term Indexing Approach for Document Categorization

Kang H. Lee[*] *Judy Kay*[*] *Byeong H. Kang*[+]

[*]Basser Dept of Computer Science
University of Sydney
N.S.W., 2006, Australia
{kangl,judy@ cs. usyd. edu.au}

[+]School of Computing
University of Tasmania
Hobart, Tasmania, 7001, Australia
bhkang @ utas. edu. au

Abstract

A Keyword Association Network (KAN) is the network of keywords extracted from a collection of documents. In this network, the relationship between keywords is represented by a confidence value. It is argued in this paper that the semantics and importance of a word can be more clearly and accurately measured by making use of other words that are co-occurring in a given document. The term frequency used for measuring the importance of terms in most document categorization methods ignores this important aspect. A KAN is constructed on the basis of co-occurring terms in documents. If two tenns appear more than a certain number of times in the same documents, they are considered as having close relationship. This paper proposes using KAN as a basis for finding informative keywords and using a confidence value in the process of document categorization. The process of constructing and application of KAN for document categorization is presented and the performance comparison with a typical statistical single-term document categorization algorithm - TFIDF classifier - will be shown. The experimental results show that KAN gives significant benefits.

Keywords Document Categorization, Machine Learning, Statistical Multi-term indexing, Semantic-Meaning.

1 Introduction

Term indexing, sometimes known as feature selection or feature extraction, is concerned with extracting informative terms - keywords - from documents based on a weighting scheme. Term indexing has been studied using a growing number of statistical and machine learning techniques and is a crucial part of both document categorization and recommendation systems for textual information. For example, the adjusting parameter (5) extracted terms and their weights are used when calculating the distance or similarity between two documents.

Based on the weighting scheme, term indexing can be broadly grouped into the following three categories: statistical, probabilistic, and information theoretic [7]. In statistical weighting schemes, frequently occurring terms in documents are regarded as informative terms and the term frequency *(TF)* is assigned to term weight. It is also widely recognized that terms which occur in only a few other documents are more informative than ones that appear in many. This consideration results in introducing the inverse document frequency *(IDF).* These two are combined into a single value, called *TFIDF* [19] and this is widely used for document categorization [8, 11, 12, 13, 22]. Probabilistic weighting schemes are also commonly used for term indexing. When applying such a weighting scheme to document categorization [3, 8, 10], the joint probability of term and category is computed from the training document set and used for the term weights to estimate the probability of a category given a document. Information theoretic methods are based on information gain [15]. In this method, the terms that are concentrated in particular documents are considered as informative terms by measuring signal-noise ratio. In this sense, it operates in a similar way to the *IDF* of statistical weighting schemes.

Term indexing is also categorized into single and multi-term indexing. In single-term indexing on which most previous research has been focused, term weighting schemes are applied to one word without considering the relationship between words. On the other hand, a different point of view on extracting informative terms is that a more accurate and precise meaning of a term can often be identified when looking at other terms in a document. For example, let

Proceedings of the 5th Australasian Document Computing Symposium,
Sunshine Coast, Australia,
December 1, 2000.

us consider the similarity between two documents where 'apple' appears in both documents. If one document is in the computer category and the other is about fruit, the term 'apple' has a different meaning in each document. If we adopt a single-term indexing scheme, the similarity value may be high and. as a result, two documents may be incorrectly considered as being similar. A multi-term indexing scheme is intended to solve the above problem by designing term weights to include information relationships between terms. Latent Semantic Indexing (LSI) [2] is an example of the multi-term indexing methods. It uses word relationships by coalescing terms which have similar meanings and has been successful in reducing the dimensionality in information retrieval area.

This paper introduces a statistical multi-term indexing scheme, the Keyword Association Network (KAN). In KAN, each word is connected with other words if they are assessed as being related and a confidence value [1] is used for measuring relationship between two words. The effectiveness of KAN for term indexing has been evaluated in document categorization. From the training and test documents in each category, our approach is to build a KAN using training data in each category and use the confidence value on the term weight when calculating similarities of test documents with categories. This approach is compared with a TFIDF classifier [16] that is one of the most widely used statistical single-term indexing approaches for document categorization.

The rest of this paper is organized as follows. Section 2 describes TFIDF classifiers and their problems. In section 3. we explain the process for building a KAN and applying it to document categorization. Experimental results are presented in section 4 to support our claim. Section 5 concludes and presents future work.

2 TFIDF classifier: A statistical single-term indexing approach

Static document categorization is the problem of assigning predefined categories to the test documents [21]. By contrast, dynamic document categorization, also called clustering, is the problem of classifying and categorizing a set of documents by grouping them [5, 12]. In this paper, we focus on the area of static document categorization.

A number of statistical learning methods have been applied to this problem in recent years. There are, in common, three main issues in applying algorithms to the document categorization problem: (1) What representation method should be used (2) How the large number of words dealt with in a representation and (3) Which algorithm is used. In the following sub-section, we look at TFIDF classifiers in terms of these issues and the problems these have for each aspect.

2.1 Document representation and the high **dimentionality** problem

To apply a learning algorithm, the first step is to transform text documents into a representation that is suitable for the algorithm to perform the categorization task. Information retrieval research suggests that words work well as representation units and that their position in a document is not very important for performance [9]. This leads to the bag-of-words representation that is widely used for document representation. However, when dealing with semi-structured documents (web documents and news articles), there is some work that uses additional information such as position, tag, or hyperlinks [4, 6]. In the bag-of-words representation, each distinct word has its frequency in a given document as a weight.

The problem in this representation is that it leads to high-dimensional word spaces. Frequently used approaches to reduce the number of different words are to use a 'stop-list' containing common words and apply a language specific stemming algorithm. Through pruning the infrequent and/or very frequent words, the size of the word space can be further reduced. However, it is noted in [9] that many irrelevant words that exist even after applying above approaches cause overfitting in measuring similarities between test documents and predefined categories. So, there is a need for a new representation method to further reduce the number of unimportant words or their influence in calculating similarities and. as a result, to improve the accuracy of document categorization.

2.2 TFIDF classifier

This classifier is based on the Rocchio relevance feedback algorithm [16]. Its major heuristic is the TFIDF word weighting scheme. Due to its various heuristic components, there are a number of similar algorithms corresponding to the particular choice of these heuristics. This sub-section describes two heuristics, the word weighting scheme and the similarity measuring method.

A TFIDF classifier builds on the following representation of documents, called the vector space model [18]. Each document D is represented as a vector $d = (v_1, v_2, \ldots, v_n)$. Here, each $v_i(d)$ is the weighting value of /th word in document D and it is calculated as a combination of two weighting schemes, TF(i, D) and IDF(r'). The term frequency TF(i, D) is the number of times the ith word occurs in document D and the inverse document frequency, IDF(z'), can be calculated as follows:

IDF(7) = log[N/DF(/)]

- DF(i) is the number of documents in which /th word occurs at least once.
- N is the total number of documents.

$pv_c = [ax\ (n_c)^{-1} \times \sum_{d \in c} d] - [\beta \times (N-n_c)^{-1} \times \sum_{d \notin c} d]$

- α and β are adjustive parameters for positive and negative examples.
- n_c is the number of documents in the category C and N is the total number of documents in the training set of documents.
- d is the vector for document D.

Set of distinct words: W = {apple, windows, computer. web, www. file}

Set of documents (D)
d_1 apple, windows, computer
d_2 windows, computer
d_3 apple, computer, web, www
d_4 file. web, www

$\{F_1\}$

Word set	Support
apple	2
windows	2
computer	3
web	2
www	2

$\{CF_2\}$

Word set	Support
apple, windows	1
apple, computer	2
apple, web	1
apple, www	1
windows,computer	2
windows, web	0
windows, www	0
computer, web	1
computer, www	1
web. www	1

$\{F_2\}$

Word set	Support
apple, computer	2
windows, computer	2
web, www	2

Figure1: Example.

Then, the weight of /th word, $v_i(d)$, in document D is TF(/, D) x IDF(/). This weighting scheme says that a word is an important indexing term for document D if it occurs frequently. On the other hand, a word which occurs in many documents is rated as a less important indexing term due to its low inverse document frequency. Because document lengths may vary widely, a length normalization factor is applied to the term weighting function. A weighting equation that is used in this experiment is given in [23].

$$v_i(d) = \frac{[\log(TF(i, D)+1.0) \times IDF(i)]}{\sqrt{\sum_{i=1,n} [\log(TF(/, D)+1.0) \times IDF(/)]^2}}$$

A prototype vector pv_c for a category C is prepared by summing the vectors of the positive documents as well as those of the negative documents and, then calculating a weighted difference for each.

The similarity value between a category and a new document is obtained as the summation of the inner products between corresponding term vectors.

The problem in this classifier is that it is very sensitive to the number of irrelevant words, because all the words participate equally in the calculation of similarity. When the discriminating words are only a small subset of the whole word set, even two documents with identical values on these discriminative words may not be considered as near neighbors. Other statistical single-term based classifiers, such as k-NN classifier [13], have the same problem.

3 Keyword association network for document categorization

The reason for proposing a new indexing scheme is to give solutions for the following two important

```
F_1: Frequent 1-word sets; CF_2: Candidate 2-word sets:
F_2: Frequent 2-word sets

for all d ∈ D do
  for all w ∈ W do
    if w_i exists in d
        w_i.count++;
F_1 = { w_i ∈ W | w_i.count >= minimum support}

when F_1 = {w_1, w_2, .... w_k}
CF_2 = 0;
for all w ∈ F_1 do
  for (i = 1; i < k; i++) do
    CF_2 = CF_2 + {(w_i, w_{i+1}),(w_i, w_{i+2}),.... (w_i, w_k)}

for all d ∈ D do
    for all cf ∈ CF_2
      if two words in cf exists in d
        cf_i.count++;
F_2 = { cf ∈ CF_2 | cf_i.count >= minimum support }
```

Compute uncommon words which satisfy user specified frequency (minimum support).

Generate word pairs for uncommon words.

Create list of word pairs which satisfy required frequency.

Figure 2: Algorithm for generating frequent 2-word sets.

objectives: (1) to give a word an appropriate weight according to its semantic meaning in a given document (2) to remove the influence of unimportant words which are misleading or irrelevant to categorization. For example. a word can have several meanings. and its meaning might be identified by considering other words in a document. This is why we use a confidence value when calculating term weight. Also, large numbers of irrelevant words are the underlying cause of imprecise document categorization. In the following sub-sections. we represent KAN - a new multi-term indexing scheme. and explain the process of applying it to document categorization by giving an example.

3.1 Multi-term indexing scheme: KAN

Previous work showed that it is possible to automatically find words that are semantically similar to a given word based on the collocation of words [17, 20]. KAN is constructed based on this statistical method. The degree of relationship between two words is represented by a confidence value. This measure was used in finding association rules [1] that have recently been identified as an important tool for knowledge discovery in huge transactional databases. In KAN, the confidence value is used for measuring how the presence of one word in a given document may influence the presence of another.

Let us assume that there is a set of n unique terms $\{W = (w_1, w_2, \ldots, w_n)\}$ and a set of m documents $\{D = (d_1, d_2, \ldots, d_m)\}$ Here, each $d_i = (w_1, w_2, \ldots, w_k)$ is a non-empty subset of W. The construction of KAN is based on support and confidence values. defined as follows.

Definition 1

The support of the term w_i - $SUP(w_i)$ - is the number of documents that contain w_i.

Definition 2

The confidence value of w_i to w_j - $CONF(w_i w_j)$ - is the percentage of documents which contain w_i and also have w_j, i.e., $SUP(w_i, w_j) / SUP(w_i)$.

The problem of building the network is to find two terms that satisfy a user-specified minimum support and confidence. High confidence of w_i to w_j is interpreted as indicating that the semantic meaning of w_i can be identified with the existence of term w_j.

In Figure 1, we have the set of documents from a certain category, information technology. Through the usual preprocessing steps of stemming and use of a stoplist as mentioned in Section 2.1, we can get the set of distinct words that are considered informative words. Note that the term 'file' occurs in just one document and so it is not considered at this point. With the given set of documents (D), set of unique words (W), and user specified minimum support (in this example, it is 2), the algorithm in Figure 2 finds the frequent 2-word sets, F_2, which are groups of two words occurring frequently together in the set of documents and satisfying the given minimum support. CF_2 is the candidate 2-word sets generated from the frequent 1-word sets, F_1. Figure 3 shows the network for above example. In the calculation of similarity between this category and a new document, if the document has both the words apple and computer they are considered as informative words. And, their confidence values are used for increasing their term weights in the similarity computation.

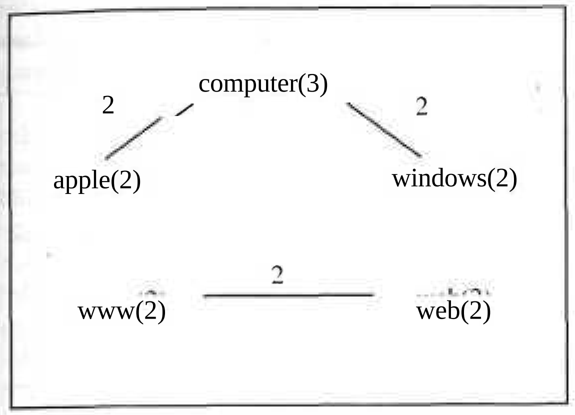


Figure 3: KAN for the example.

3.2 Applying KAN to document categorization

The most important factor in increasing the accuracy of categorization is to remove the influence of the large number of irrelevant terms that occur evenly across the categories. Those terms should be connected with many other tenns in the network. This means their confidence values are quite low compared to the confidence values of the important terms; they will significantly degrade the overall accuracy of categorization. So, we need to restrict the number of the participating tenns in the similarity measure through filtering out these irrelevant terms. To solve this problem, we use the inverse category frequency (ICF).

Definition 3

The inverse category frequency of term w_i, $ICF(w_i)$ is:

$$ICF(w_i) = [\log (C / c_i) / \log C] + \delta$$

where c_i is the number of categories in which w_i appears in the KAN, C is the total number of categories, and δ is a parameter that adjusts the impact of ICF values to term frequencies.

In a network, support of each term is replaced with the weighted support (WS), and it is used in computing the modified confinece value ($MCONF$).

Definition 4

The weighted support of term w_i, $WS(w_i)$, is:

$$WS(w_i) = SUP(w_i) / ICF(w_i)$$

Definition 5

The modified confidence value of w_i to w_j, $MCONF(w_i w_j)$, is:

$$MCONF(w_i w_j) = SUP(w_i, w_j) / WS(w_i).$$

If a term appears only in one network. its ICF value will be 1 plus the adjusting parameter value (δ). So. its weighted support (WS) in a network will be smaller than its original term frequency (TF) and. as a result. its confidence values to other terms in a network become greater than the original confidence values. In the other extreme situation, consider a term that appears in all given networks. Its ICF value will be same as the adjusting value since its numerator. log (C / c_i), in the definition 3 will be zero. So. its WS becomes much greater than its term frequency and its modified confidence values to other terms become much smaller. In this way, we can greatly decrease the impact of irrelevant terms to the similarity calculation.

In the similarity computation between a category and a new document, the weight (TFIDF) of each term in the category is increased by the value of multiplying its original term weight by its total confidence value. The total confidence value for each term is obtained by finding all the links from other terms in the KAN. summing up all the modified confidence values which are greater than the threshold. Thus. the similarity value between a document D and a category C. which are represented by the vectors of the form $(d_1, d_2, \ldots, d_k)$ and $(c_1, c_2, \ldots, c_k)$ respectively, is computed as follows:

$$Sim(D, C) = \sum_{i=1,k} \{[c_i + (c_i \times \sum_{j=1,k \& j \neq i} MCONF(w_i w_j))] \times d_i\}$$

d_i: the weight of ith word in the vector of document D.

c_i: the weight of ith word in the vector of category C.

w_i, w_j: the ith and jth words in document D.

4 Experiments

Experiments were conducted to find out how well KAN performs in document categorization. The performance of KAN is compared with a TFIDF classifier in the following data set.

Category	Number of Training data	Number of Test data
earn	2709	1066
acq	1488	722
money-fx	460	222
grain	394	179
crude	349	215
trade	337	177
interest	289	133
wheat	198	89
ship	191	103
corn	159	63

Table 1: Number of training and test data in each category.

4.1 Charactreristics of data set

The data set is the Reuters-21578 text categorization test collection Distribution 1.0. In this collection, each article does not contain any meta-information (hyperlinks or tags) that could be used for extracting more important terms. Instead of analyzing all 135 categories, we select the ten most frequent categories and split the articles into training and test set according to the Modified Lewis Split as shown in Table I. For the preprocessing steps, we applied a stop-list and Porter's stemming algorithm [14] to the articles. Then we take terms that appear more than twice in a document as informative tenns.

4.2 Experimental results

Table 2 shows the accuracy of the two classifiers in each category. To build KAN for each category, the larger number between the integer number of 2% of training data and 5 was set as minimum support and minimum confidence = 50 was used for all categories. Also, when computing the ICF value, the adjusting parameter (5) was set to 3 for terms which appear in only one category and 0.05 for other terms. Comparing KAN and TFIDF classifiers, KAN tends to work better in all categories. The accuracy values of the two classifiers are much lower than expected in grain and this is due to the fact that grain has similar characteristics with corn and wheat.

The accuracy in each category seems to be affected by the existence of similar categories that have many keywords in common. So, the most important task in document categorization is to find out the informative and unique terms in each category and give high weights to them. Our multi-term indexing method, KAN and its new weighting scheme achieve comparable performance over TFIDF in this unfavorable experimental situation.

Category	TFIDF(%)	KAN(%)
earn	91.5	96.2
acq	67.7	84.6
money-fx	73.8	81.1
grain	42.5	50.3
crude	76.9	79.1
trade	90.8	98.3
interest	68.2	85.7
wheat	68.6	96.0
ship	60.2	74.8
corn	90.0	95.0

Table 2: Accuracy of classifiers in each category.

5 Conclusion and future work

In most single-term indexing methods, terms are indexed syntactically and the frequency is the only data showing their importance in the documents. It has been noted that their lack of ability to capture the semantic meanings of terms has reduced their performance in textual information systems.

In a given document, the exact meanings of some words can only be identified in relation to other words. Also, their meanings tend to change in accordance with other co-occuning words. KAN is a multi-term indexing method and designed to capture a certain level of the semantic meanings of words.

In this paper, this new multi-term indexing scheme called KAN was applied to document categorization. The expected advantages of applying KAN in the task of document categorization are: (1) unlike single-term indexing methods, it becomes possible to give weights to terms according to their semantic meanings in a given document. This is achieved by using the confidence value - $CONF(w_i w_j)$ - between two terms, and (2) the influence of a huge number of unimportant terms can be further removed by increasing their supports and is handled by dividing supports with their inverse category frequency *(ICF)* values, i.e. the use of $MCONF(w_i w_j)$. Our experiments indicate that our approach is more successful than a typical statistical single-term classifier (TFIDF).

In future work, KAN will be evaluated to establish how quickly its accuracy increases. It is also planned to build KAN at the sentence or paragraph level. When the document is quite long, it seems to be desirable to consider the sub-section of a document as a basis unit for building KAN. Through the experiments, it was identified that the perfomance of KAN is very sensitive to the parameters, such as 8, the minimum support, and the minimum confidence. The more research will be done to determine the optimum parameter values automatically in each category.

We have also noted that the application of KAN to other areas could be promising. For example, in textual recommendation systems, each user's profile built by KAN will represent her/his interests more accurately. Also, unlike single-term indexing methods KAN could be used to expand user's query in information retrieval systems. The KAN's usefulness in the selection of important words and its relationship information among keywords make it possible for the retrieval systems to suggest other keywords that are highly related to the user's input query.

References

[1] A. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A.I. Verkamo. Fast Discovery of Association Rules. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smith. R. Uthurusamy, editors. Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press, pages 307-328, 1996.

[2] M.W. Berry, S.T. Dumais, and G.W. O'Brien. Using Linear Algebra for Intelligent Information Retrieval. SIAM Review, Vol. 37, No. 4, pages 573-595, 1995.

[3] L.D. Baker and A.K. McCallum. Distributed Clustering of Words for Text Classification. ACM SIGIR98, 1998.

[4] W.W. Cohen, Learning to Classify English Text with ILP Methods. Workshop on Inductive Logic Programming, Leuven. September, 1995.

[5] P. Cheeseman and J. Stutz. Baysian Classification (Autoclass): Theory and Results. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smith, R. Uthurusamy, editors. Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press, pages 153-180, 1996.

[6] J. Furnkranz. Exploiting Structural Information for Text Classification on the WWW. In D. J. Hand, J. N. Kok, M. R. Berthold (eds.), Advances in Intelligent Data Analysis: Proceedings of the 3rd Symposium(IDA-99), Amsterdam, Netherlands. Lecture Notes in Computer Science 1642, Springer-Verlag, pages 487-497, 1999.

[7] V.N. Gudivada, V.V. Raghavan, W.I. Grosky, and R. Kasnagottu. Information Retrieval on the World Wide Web. IEEE Internet Computing, pages 58-68, September-October 1997.

[8] T. Joachims. A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. In Proceedings of the 14th International Conference on Machine Learning ICML'97, pages 143-151, 1997.

[9] T. Joachims. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In Machine Learning: ECML-98, Tenth European Conference on Machine Learning, pages 137-142, 1998.

[10] D. Lewis and M. Ringuette. A Comparison of two learning algorithms for text categorization. In Third Annual Symposium on Document Analysis and Information Retrieval, pages 81-93, 1994.

[11] D.D. Lewis, R.E. Schapire, J.P. Callan. and R. Papka. Training Algorithms for Linear Text Classifiers. In SIGIR 96: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 298-306, 1996.

[12] J. Moore. E. Han. D. Boley, M. Gini, R. Gross, K. Hastings. G. Karypis, V. Kumar, and B. Mobasher. Web Page Categorization and Feature Selection Using Association Rule and Principal Component Clustering. In 7th Workshop on Information Technologies and Systems. Dec. 1997.

[13] S. Okamoto and K. Satoh. An Average-Case Analysis of k-Nearest Neighbor Classifier. In Proceedings of the First International Conference on Case-Based Reasoning, pages 243-264, 1995.

[14] M.F. Porter. An Algorithm for Suffix Stripping. Program, Vol. 14. No. 3, pages 130-137, July 1980.

[15] J.R. Quinlan. C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993.

[16] J. Rocchio. Relevance Feedback in Information Retrieval. In G. Salton, editor, The SMART Retrieval System: Experiments in Automatic Document Processing, Prentice-Hall Inc., 1971.

[17] G. Ruge. Experiments on Linguistically Based Term Associations. Information Processing & Management, 28(3), pages 317-332, 1992.

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

[19] G. Salton and C. Buckley. Term Weighting Approaches in Automatic Text Retrieval. Information Processing and Management, Vol. 24, No. 5, pages 513-523, 1998.

[20] S. Sekine, J. Carroll, A. Ananiadou, and J. Tsujii. Automatic Learning for Semantic Collocation. Proceedings of the Third Conference on Applied Natural Language Processing, ACL. pages 104-110, 1992.

[21] Y. Yang. An Evaluation of Statistical Approaches to Text Categorization. Journal of Information Retrieval, Vol. 1, No. 1/2, pages 67-68, 1999.

[22] T. Yavuz and A. Guvenir. Application of k-Nearest Neighbor on Feature Projections Classifier to Text Categorization. In Proceedings of the 13th International Symposium on Computer and Information Sciences - ISCIS'98, U. Gudukbay, T. Dayar, A. Gursoy, E. Gelenbe (eds.), Antalya, Turkey, pages 135-142, Oct. 26-28, 1998.

[23] C. Buckley, G. Salton, J. Allan, and A. Singhal. Automatic Query Expansion Using SMART: TREC 3. The Third Text Retrieval Conference (TREC-3). National Institute of Standards and Technology Special Publication 500-207, Gaithersburg, MD, 1995.

The TREATS Approach to Reuse of Tables in Plain Text Documents.

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

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

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

Abstract.

In this paper we present the table processing approach employed by the TREATS (Table Recognition, Extraction, Analysis and Tranformation System) software toolkit.

In order to support the large variety of table layouts that appear in plain text documents, our system aims to identify the layout of cells that exist within a table and to tailor processing accordingly. This results in more effective processing than preexisting approaches that apply one general technique to all types of table.

The classification process is the key to processing and exploits a cellular automaton (CA) based approach to the identification of cell structure within a table. The input to the CA is a simple representation of the content of the source table. This is evolved via the application of intelligent transformation rules, resulting in a representation of the cell structure that exists within the table. Based on the combination of cell types that appear in this representation, the layout of the table can be determined and appropriate processing can be performed. During this processing, the content of the source table is transformed into a relational form suitable for reuse in other applications.

Keywords: Information retrieval, resource discovery.

1. Introduction.

The TREATS (Table Recognition, Extraction, Analysis and Transfonnation System) software toolkit [1,2] offers support for reuse of tabular content from a wide variety of source documents[2]. Of the supported types, tables in plain text form are by far the most difficult to reuse. Whilst other document types such as HTML and Latex contain table definitions that enable their content to be extracted via a parsing approach, with plain text the only guide to the structure of the table (the key to correct reuse) is the layout of its content. Combined with the variety of table layouts that are possible, this makes tables in plain text documents difficult to effectively and conectly reuse.

Although methods for the processing of such tables exist, we feel that in general a specific approach can only support reuse of a limited subset of the possible table layouts. In this paper, we describe the approach used by the TREATS toolkit that provides more wide ranging support by identifying and classifying table layout such that processing can be tailored accordingly.

During this discussion, we concentrate on processing the content of a table i.e. everything below the level of the column labels[3]. Whilst these labels are important to the understanding of table content and will need to be extracted and in some cases transformed[4] to enable reuse, they do not offer any indication of the cell structure of the table.

The remainder of a table is concerned with the presentation of data and consists of two components, the stub and the body [3]. In general, the stub is the leftmost column of a table and may contain either a simple column of entries or in some cases where grouping exists, nested entries that indicate the hierarchy within the data. This hierarchy may also be indicated through the use of spanning cells and where this occurs, we extend the definition of the stub to cover all columns that contain spanning cells involved in defining the hierarchy. Spanning and nested cells are discussed in more detail in section 3.1. The body

[1] With support from EPSRC and BT (Case Studentship).
[2] Currently, plain text, HTML and Latex are supported.
[3] The area where labels are displayed is often known as the boxhead.
[4] Processing of label structures is addressed elsewhere [1].

Maker	Model	Esize
Ford	Fiesta	1.1
Ford	Escort	1.3
Vauxhall	Astra	1.2
Vauhall	Carlton	1.6

Type I

Maker	Model	Esize
Ford	Fiesta	1.1
		1.3
	Escort	1.3
		1.6
Vauxhall	Astra	1.2
		1.3
	Carlton	1.6
		2.0

Type II

	Price	Insurance Group
Ford		
Fiesta		
1.1	8000	38
1.3	8650	35
Escort		
1.3	9000	35
1.6	9750	32

Type III

Model	Axle Type
Fast Eddy Shooter	Fixed wooden axle
Henrys Viper	Ball bearing transaxle

Type IV

Figure 1: Illustration of Table Layouts.

of the table consists of all columns not involved in the stub.

2. Complexities of Processing Tables in Plain Text Documents.

When designing a table in plain text, an author has few constraints on how its content should be laid out, beyond accepted styles for table content [3,4]. The result of this is that a variety of table layouts are commonly encountered within plain text documents. Because of this flexibility, existing table processing approaches based on white streams [5,6] and bounding boxes [7.8] may only be effective for certain types of table layout. The reason for this is that these techniques employ a single overall approach that handles all tables in the same manner. Unfortunately, this is not always effective since tables with different types of layout may need to be handled in different ways to conectly extract their content.

Thus, we feel that in order to effectively process an arbitrary plain text table, tailored processing will be required. To achieve this, the layout of the table must be identified whereupon appropriate processing can be employed. To enable this type of approach, through a survey of a large range of tables from various sources[3] we have identified five classifications of table layout. These are based on the different types of layout that may appear within the stub and the body of a table:

- Type I - tables with simple cells that span only a single row and column.
- Type II - tables where stubs contain spanning cells.
- Type III - tables where stubs contain nested entries.
- Type IV – tables whose entries span multiple lines.
- Type V - hybrid layouts that combine elements of the other classifications but don't fit into any particular one.

Stub layout		Body Layout	Classification
Single line onirics		Single line entries	Type I
Spanning cell	with single line entries	Single line entries	Type Ha
	with multi-line entries	Single line entries	Type IIb
	with single-line entries	Multi-line entries	Type IIc
	with multi-line entries	Multi-line entries	Type IId
Nested cell	with single line entries	Single line entries	Type IIIa
	with multi-line entries	Single line entries	Type IIIb
	with single line entries	Multi-line entries	Type IIIc
	with multi-line entries	Multi-line entries	Type IIId
Single line entries		Multi-line entries	Type IVa
Multi-line entries		Multi-line entries	Type IVb
Multi-line entries		Single-line entries	Type IVc

Figure 2: Layout Classifications.

Examples of these layouts can be seen in Figure 1. Within each classification, a number of alternative layouts can result if entries that span more than one line appear in the stub, the body or both. The table in Figure 2 shows all possible layouts that are available within Type I-IV classifications.

[3] Numerous engineering and medical texts were consulted during the survey.

3. The TREATS Approach to Table Structure Processing.

The approach employed in the TREATS toolkit is to firstly classify the source table, then apply appropriate processing to enable the transformation of its content into a form suitable for reuse.

Tables are located within source documents using a method that exploits the patterns of whitespace and data types that appear within tables [1]. The pattern of whitespace between columns of entries is generally consistent for each line of the table body, as are data types for each entry within a column. By examining the patterns of whitespace and data types that appear for each line of the source document, it is possible to locate and extract areas of text that contain tables.

3.1 Classifying Table Layout.

In order to classify table layout, TREATS attempts to locate any of four basic 'table elements' within a source table. The combination of such elements within a column and the position of the column can be used to identify the layout of the table based on the classifications presented in Figure 2. Within a table, the following table elements may appear.

3.1.1 Simple (single-line) Cells.

The most basic type of cell, this consists of a single line of text and can appear anywhere in a table.

3.1.2 Multi-line Cells.

Multi-line cells contain entries that span over a number of lines (Figure 3). By their nature, multi-line cells will only contain string type entries. In order to locate multi-line cells our approach relies on the fact that either the start line or all continuation lines will be indented[6].

```
This is a Multi-line      This is not!
 Cell
```

Figure 3: A Multi-line Cell.

3.1.3 Spanning Cells.

Spanning cells appear where grouping relationships occur in a table, as in the first two columns in Figure 4. They are used to indicate groupings of related data, where the relationship is indicated by the number of rows over which the cell spans. Spanning cells can be nested to form hierarchies. In most cases, these appear on the left of a table and are considered to represent the stub, although in some hybrid layouts they may appear within the body of the table. They may contain entries that span one or more lines. As with multi-line cells, entries that span more than one line are generally indicated through the use of indents.

```
Animals   Cats   Persian
                 British Blue
          Dogs   Collie
                 Alsatian
```

Figure 4: Spanning Cells.

3.1.4 Nested Cells.

Nested cells contain entries that are nested to form a classification hierarchy as illustrated in Figure 5. These cells generally appear in the stub[7] and may contain single or multi-line entries. Nested cells are characterized by multiple levels of indenting, where the level of indent indicates the level at which an element belongs in the hierarchy. Notice that there is a relationship between nested cells and spanning cells and that they can be used interchangeably (e.g. Figure 5 is equivalent to Figure 4).

```
Animals
 Cats
  Persian
  British Blue
 Dogs
  Collie
  Alsatian
```

Figure 5: Nested Cells.

3.2 The Classification Process.

The table classification process involves the following steps:

i) Determine the alignment and data types of the entries in each column.
ii) Identify the table elements in each column - indicating the cell structure,
iii) Classify layout.

Before any classification processing can take place, the column structure of the table is deduced using an extension to a projection profile approach that exploits vertical streams of white space between columns [1].

[6] If they are not, it is impossible to determine the existence of multi-line entries automatically.

[7] In rare cases, where hybrid tables occur, they may appear elsewhere in the table body.

3.2.1 Determining Alignment and Data Types.

The alignment and data types of entries in columns directs processing in a number of ways. The alignment of entries within a column provides clues to logical structure embedded within the content and is an essential guide to the recognition of nested and multi-line entries within a table. As a number of alignments are possible it is essentia] to be able to uniquely identify alignment to ensure that table elements are correctly recognised. A number of different types of alignment for column entries can be identified:

- Left aligned - the most common type of alignment.
- Right aligned - often seen where integers are present within a table
- Centred.
- Nested.
- Multi-line – both indenting of continuation lines (the most common layout) and indenting of the first line of an entry (paragraph style) are supported[8] by TREATS.

Data types are exploited to overcome layout conflicts that may occur. For example, it is possible that a column of right aligned integers may have a layout that is identified as possibly being a column with multi-line entries. Since only string type entries can form multi-line entries, inconect processing of ambiguous layouts can be avoided by data type checking.

3.2.2 Identifying Table Elements.

Identifying table elements is achieved by examining the layout of content within columns and matching this to the layout that would be expected for these elements. In tables with a relatively simple structure, this can be done by examining the patterns of layout within a single column. In tables with more complex layouts however, due to ambiguity that may occur within a single column, the layout of entries in neighbouring columns must also be considered to ensure that elements are correctly identified. For this reason, TREATS employs a Cellular Automaton (CA) [9,10] based approach to the identification of table elements.

A Cellular Automaton is an array of identically programmed automata or 'cells' which interact with each other. At regular intervals, the content of these cells is evolved. Evolution is directed by a number of transformation rules, where the evolution of a cell is controlled by its state and those of its neighbouring cells.

To determine cell stiucture, a simple representation of the table is evolved via the application of intelligent transformation rules, resulting in a representation that clearly indicates the elements that exist within the table. This process has three steps:

i) Generate initial table representation.
ii) Evolve cell structure through repeated application of transformation rules.
iii) Extract table element/cell stiucture information from the evolved representation.

Generating the Initial Representation.

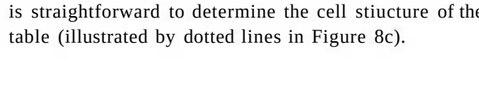
The first stage of the process is to generate a representation of the content of the table within the CA. This is modelled in a simple 2-D array which contains tokens that represent the layout of the entries within the table. Each line of text within each column is represented as its own cell. To illustrate this, consider Figure 8b which shows the initial representation for the table in Figure 8a. The choice of token is guided by the layout and content of a cell and the alignment information deduced in the previous phase. The tokens used in the representation are summarised in Figure 6.

Token	Description
s	A simple cell with no formatting[9]
e	An empty cell
i	A cell with an indented entry - indicating inclusion in a multi-line entry
n?	A cell involved in a nesting hierarchy. ? indicates the level of nesting, starting at 0

Figure 6: Tokens in Original CA Representation.

Evolution of Table Representation.

The initial representation is evolved by the repeated application of a number of transformation rules. These rules are applied to every cell in the CA until no further transformations occur (i.e. the CA becomes stable). An example of two rules used to evolve multi-line cells can be seen below:



The table in Figure 7 summarises the tokens that are used in the transformation rules and consequently appear in the evolved table representation.

Token	Description
es	Empty cells that are involved in a spanning cell
c	Continuation sections of multi-line entries
nc	Continuation sections of multi-line entries appearing within nested stubs
v	Void cells - empty cells that appear opposite classifications of a nested stub and are not involved in spanning cells

Figure 7: Tokens Involved in Transformation Rules and Resulting CA Representation.

Extracting Table Element/Cell Structure from the Evolved Representation.

Figure 8c shows the result of transforming the initial representation of the table in Figure 8b. From this, the cell structure is clearly defined by the tokens defined in Figures 6 and 7. Determining which table elements appear in each column is achieved using a pattern matching approach e.g. s followed by one or more c's indicates the existence of a multi-line cell. From the positions and sizes of these CA elements, it is straightforward to determine the cell stiucture of the table (illustrated by dotted lines in Figure 8c).

3.2.3 Classifying Layout.

From the information about column alignment and cell structure the layout of a table is classified through the application of niles based on the table characteristics presented in Figure 2. An example of one of these rules is:

```
If stub contains nesting and exhibits multi-
line entries
  if the body contains multi-line entries
   classification is Type IIId
  else
   classification is Type IIIb
```

3.3 Processing and Transformation of Table Content.

Once the cell structure of a table has been determined, the content of the source table is transformed into a relational fonn to enable its reuse. This processing requires two actions:

i) Multi-line entries are replaced by an equivalent single line entry.
ii) For Type II and III layouts, nested and spanning cells must be transformed into an appropriate form.

Multi-line entries must be processed first as they may appear within nested and spanning cells. Multi-line entries require only simple transformation involving the concatenation of the content of all line segments over which the cell concerned spans.

Since nested and spanning cells cannot be supported in a relational table, where such layout occurs content must be transformed. For nested entries, the nesting must first be expanded so that a column exists for each level of nesting (as with the equivalent content where spanning cells are used).

Type	Benign/ Malignant	Tumor
Bone forming	Benign	Osteoma
		Osteiod osteoma
Marrow tumor	Malignant	Ewing's sarcoma
		Lymphoma (see pages 11.61 - 11.66)
		Myleoma (see pages 11.67 - 11.69
Synovial tumor	Benign	Pigmented villonodular Synovitis
	Malignant	Synovial sarcoma

(a) The Source Table

(b) Initial CA Representation

(c) Evolved CA

Figure 8: A Table and its Associated CA Representations.

[8] In some cases it is impossible to determine which layout exists. Where this is the case, the user is required to determine which type of layout is used.

[9] Note that non-left aligned entries are considered simple entries in this representation, as they are not involved in multi-line entries.

Finally, repeated entries are introduced. Only this final stage is required during the processing of spanning cells. This process is illustrated in Figure 9, where Figure 9a shows an example of a nested stub (see Figure 4 for the equivalent with spanning cells) and Figure 9b shows the equivalent relational fonn, with reintroduced repeated entries indicated by italics.

(a) Nested stub. (b) Transformed stub.

Figure 9: Transformation of a Nested Stub.

4. Conclusion.

By classifying table layout and tailoring table processing accordingly, the TREATS approach to processing plain text tables can effectively support a large variety of table layouts. It has been tested on many of the examples used in the aforementioned layout classification survey and has proved to be highly effective.

We have identified a small number of limitations of our approach, where processing is affected by ambiguous layouts due to empty cells resulting from missing entries and lack of indentation to indicate where multi-line entries appear. Missing entries result in the incorrect formation of spanning cells and lack of indentation means that multi-line entries cannot always be formed. Unfortunately, there is no way to overcome these limitations automatically, due to conflicts with other types of layout. These problems can only be overcome through user interaction, but are exceptional.

Finally, whilst we have not considered hybrid tables in this paper, support for the most commonly appearing forms is available in the TREATS toolkit through suitable combination of the techniques we have described.

5. References.

[1] L. E. Hodge. W. A. Gray and N. J. Fiddian. Effective Reuse of Textual Documents Containing Tabular Information. *Proceedings of the 4th Australasian Document Computing Symposium (ADCS 99)*. Coffs Harbour. NSW. Australia, December 1999. pp 47 - 53.

[2] L. E. Hodge, W. A. Gray and N. J. Fiddian. A Toolkit to Facilitate the Integration of Tabular Information in Textual Documents with Database Applications. *Proceedings of the 4th Multiconference on Systemics, Cybernetics and Informatics (SCI 2000)*, Orlando. Florida, USA, **July** 2000.

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

[4] R. J. Beach. Tabular Typography. *Proceedings of the International Conference on Text Processing and Document Manipulation*, University of Nottingham. April 1986. pp 18- 33.

[5] T. Pavalidis and J. Zhou. Page Segmentation by White Streams. *Proceedings of the International Conference on Document Processing (ICDAR '91)*. Saint Malo. Fiance. 1991. pp 945-953.

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

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

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

[9] J. D. Farmer. T. Toffoli and S. Wolfram, editors. *Cellular Automata: Proceedings of an Interdisciplinary Workshop*, Los Alamos, New Mexico, March 7-11, 1983.

[10] Stephan Wolfram. *Cellular Automata and Complexity: Collected Papers*. Addison-Wesley, 1994.

Recovering Structure from Unstructured Web-accessible Classified Advertisements

Richard Cole, Peter Eklund and Age Strand
School of Information Technology, Griffith University
PMB 50 Gold Coast MC
QLD 9726. Australia
{r.cole,p.eklund}@gu.edu.au, mstrand@hotmail.com

Abstract

This paper describes a research prototype system called RFCA for structuring Web-accessible rental classified advertisements based on semantic content. A hand crafted parser is used to extract various facets of the rental property being advertised including amongst others; member of room, type of garage. dwelling type (unit, house. or high rise apartment), price and contact details. The performance of the parser is measured in terms precision and recall by comparing its output to that of human expert.

The structured information once extracted is stored in a relational database and users searching for rental properties are presented with a graphical organisation of rental properties according to predefined themes. The overall result is a suite of tools for extracting. cleaning, structuring, and visually querying/browsing collection of web-accessible rental advertisements.

The mathematical and. methodological foundation for the graphical organisation of the structured information is provided by formal concept analysis. Using formal concept analysis each property is understood to be. an object possessing attributes with attribute values. The data is then conceptually organised via concept lattices dynamically according to a pre-definedconceptualscales. The. concept lattice organises rental properties into conceptual groupings. The, user then has the opportunity to view the attributes of all properties in a grouping as well as navigate back to the source advertisements.

The. interface is delivered over the web using a CGI interface and dynamic creation of image and image maps. The. ideas presented are general enough to be relevant to other web-accessible unstructured text sources.

1 Rental Formal Concept Analysis

Many newspapers hold large keyword indexed free-text collections of classified advertising on the Web and these can be searched on-line. The intention with this work is to demonstrate how such data can be value-added by extraction. cleaning and structuring. A structured database derived from free-text classifieds can then be browsed effectively. We argue that a browsing interface that structures the presentation of classifieds, related to a particular purpose (such as rental classifieds), can facilitate retrieval. More specifically, we maintain that a browsing interface using formal concept analysis gives a sense of the way that attributes within the free-text sources are distributed across the text collection. something that a keyword-based search interface cannot do.

Formal Concept Analysis (FCA) [13] has been developed during the last twenty years and successfully applied to data analysis and knowledge processing [15]. The Mathematics of FCA has been carefully described in Ganter and Wille [9] and the basic details of the theory are omitted here for brevity. There have been a number of examples of FCA applied to information retrieval and filtering [10, 1] including our own work [3. 4]. In these systems the main difficulty is attribute identification front texts. In the medical and email domains in which have worked [8. 2. 5]. objects are easily identified since they correspond to documents: typical stored as a single file. In the case of Web-accessible rental classifieds the extraction task is complicated by object recognition: several rental classifieds often appear in the same advertisement and are grouped by location. (trice or the number of bedrooms. An example of such a problematic advertisement is illustrated in Fig. 1.

2 Object and Attribute Identification

For these reasons one of the first tasks of a unstructured text parser for RFCA is object recognition. disambiguating a single rental property from an advert that may list several or many properties for rent. For example, in Fig. 1, lines 3, 4, 5, 7 and 8 of the advert are individual properties which require representation as objects. In addition, there are cases where attribute recognition can also be com-

Proceedings of the **Fifth Australasian** Document Computing Symposium, Sunshine Coast, Australia, December 1, 2000.

```
FDR RENT - ARUNDEL - Phone 55948184
$300
4 Bedrm, in-grnd pool, dble garage, near
  shops and school
3 bedrm, tripple garage, immac. presented,
  close to transport
Exec. 3 Bedrm + study, pool, dble garage, all ammen.
  close to  school
$250
Leafy 3 bedrm, double garage, avail. Aug.
3 bedrm townhouse, resort fac. l.up garage,
  2 bathroom and on-suite.
Townhouse, 2 bedroom, resort fac. garage,
  near golf course and transport.
```

Figure 1: A rental classified illustrating multiple aliases for attributes (as in abbreviations such as Bedrm=bedroom), multiple objects (as rental properties described on lines 3, 4, 5. 7 and 8) in a single advert (all lines) clustered on an primary key attribute: in this case the two prices $300 and $250.

plex because multiple FCA objects may be clustered under a single attribute. For instance, in Fig. 1 $300 per week applies to the properties 3, 4 and 5 while $250 applies to properties described in lines 7 and 8.

To formalise the understanding of objects as properties having associated attributes we introduce a basic structure of formal concept analysis — the multi-valued context. A *multi-valued context* is a tuple (G, M, W, I) where G is a set of properties. M is a set of attributes. $W = \cap_{m \in M} W_m$ is a set of attribute values and $I \subseteq G \times M \times$ IF is a relation saying which rental properties have which attribute values for which attributes. The relation I is restricted so that for any rental property and attribute there is only one attribute value in I. More formally; I is a relation such that if $(g,m,w) \in I$ and if $(g, m, w_2) \in I$ then $w_1 = w_2$.

Multi-contexts are converted to single valued contexts via a mechanism called conceptual scaling. A conceptual scale for an attribute m is a triple $\mathbb{S}_m$ = (IF,,,, M_s, I_s)where M_s is a set of new binary attributes. The relation $I_s \subseteq$ IF,,, x M_s says which new attributes are indicated by which attribute values. So for example consider the conceptual scale in Figure 2. This scale introduces the new attributes *some kind of parking, double, single, garage, carport,* and *no parking.* Each circle represents a collection of real estate properties. For example the circle marked *single* represents all properties with a *single* car spot. The circle marked *garage* represents properties with a *garage* and the circle below and connected to these two circles represents properties with *single garages.*

Each circle in the conceptual scale has associated with it a fragment of an SQL query that allows

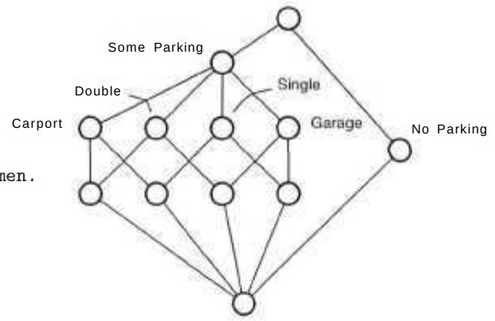


Figure 2: An example conceptual scale for parking. The scale implies that a rental property cannot have a carport that is both a double and a single.

the rental properties for that circle to be extracted from the relational database. One may notice in the conceptual scale that there is a type ordering present. For example because the circle for *garage* appears below the circle for *some parking* the set of rental properties with garages are considered a subtype of the set of properties with parking. Another way to look at this situation is to see that the type *some parking* is a super-type of the disjunction of rental properties with a single or double. garage or carport.

The concept lattice derived from the conceptual scale has the same structure as the scale hut indicates the number of rental properties classified under each circle. These numbers are calculated dynamically by interrogating the database and using the SQL fragments attached to each circle. This approach avoids having to either calculate large concept lattices or indeed try to read large lattices. It is possible to combine conceptual scales using a special diagram called a "nested line diagram" [9].

Figure 3 shows a concept lattice derived from the scale for '"Geographical position on the Gold Coast". Suburbs are clustered in the middle layer of the hierarchy and since no property in the set we considered can be in more than one suburb there are no intersections between the attributes in the third layer. For scales such as the "facilities" scale, including attributes like "close to shops", "close to transport", "close to sporting facilities" there are many instances of rental properties having a mixture the attribute. By combining two scales the user is able to see the trade-off between various facilities, cost, geographical location or number of car spaces.

Suburbs are clustered in the central layer of the diagram in Figure 3. Since suburbs are mutually exclusive sets, there are no intersections between the rental properties, the numbers of which are showing in the third layer, and the central layer. In other thematic views, such as the parking scale

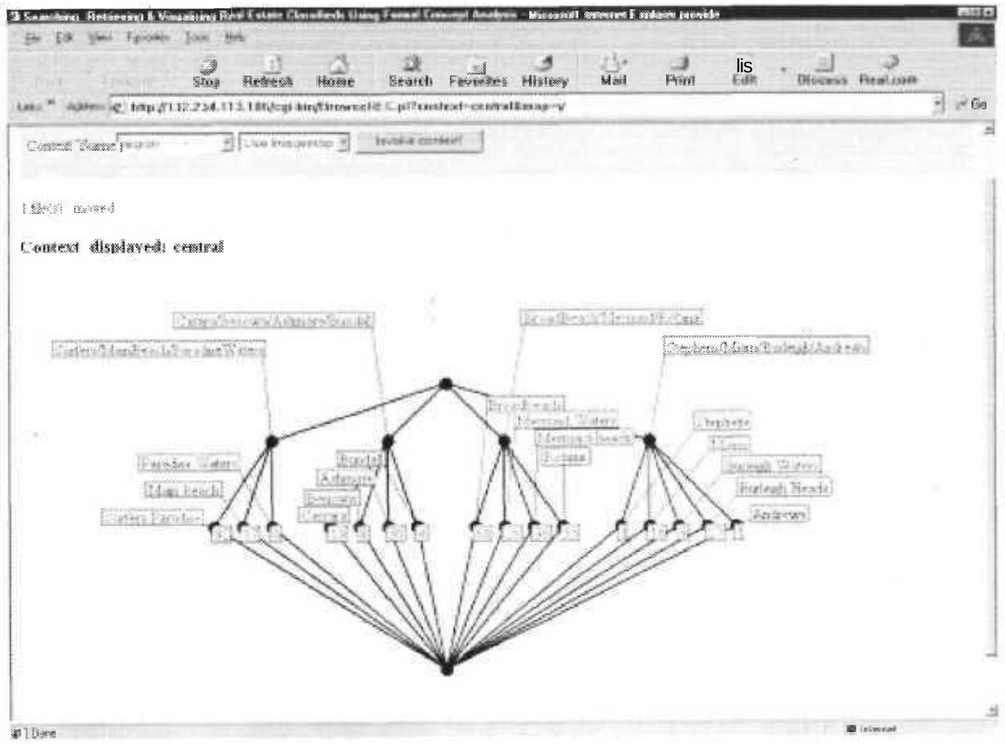


Figure 3: A concept lattice showing the breakdown of location and their geographic spread on the Gold Coast.

shown in Figure 2. there are many instances of rental properties exhibiting various attribute combinations, an example is shown in Figure 4. where the number 5 indicates the number of rental properties that exhibit the attributes double and carport. By combining two scales the user can explore the tradeoff between attributes in both scales. e.g. geographical locations and parking.

In other applications [2. 4] we have experimented with allowing the user to construct conceptual scales on the fly. However this raises questions of scalability [5] and graph layout [6]. In such a constrained task as searching and browsing rental advertisements we deem the approach outlined in the following section sufficiently flexible.

3 Browsing the Rental Classifieds

Consistent with the standard practice of formal concept analysis [9] we consider the navigation space as the direct product of the concept lattices derived from all scales. The user employs two basic operations to focus their attention at varying levels of details. The two basic operations are nesting and zooming.

Nesting allows the user to combine two conceptual scales. For example if the user is considering the concept lattice derived from the scale in Figure 2 and wants to see how the number and type of car spaces is affected by geographical area then the user can construct a nested line diagram. Such a combination is shown in Figure 5.

Zooming allows the user to restrict the set of rental properties shown in the diagram. Say for example that the user has after looking at the nested diagram in Figure 5 decided that they are interested in properties in Surfers/Main Beach with some parking. The user can zoom in by selecting the concept for "Some Parking" and "Surfers/Main Beach". In this case the next scale selected will show a concept lattice containing only the properties in Surfers/Main Beach that have some parking mentioned. The user also has the possibility to anti-zoom. That is remove the previous zooming restriction. By composing a series of zooms, anti-zooms and nestings the user is able to organise the properties

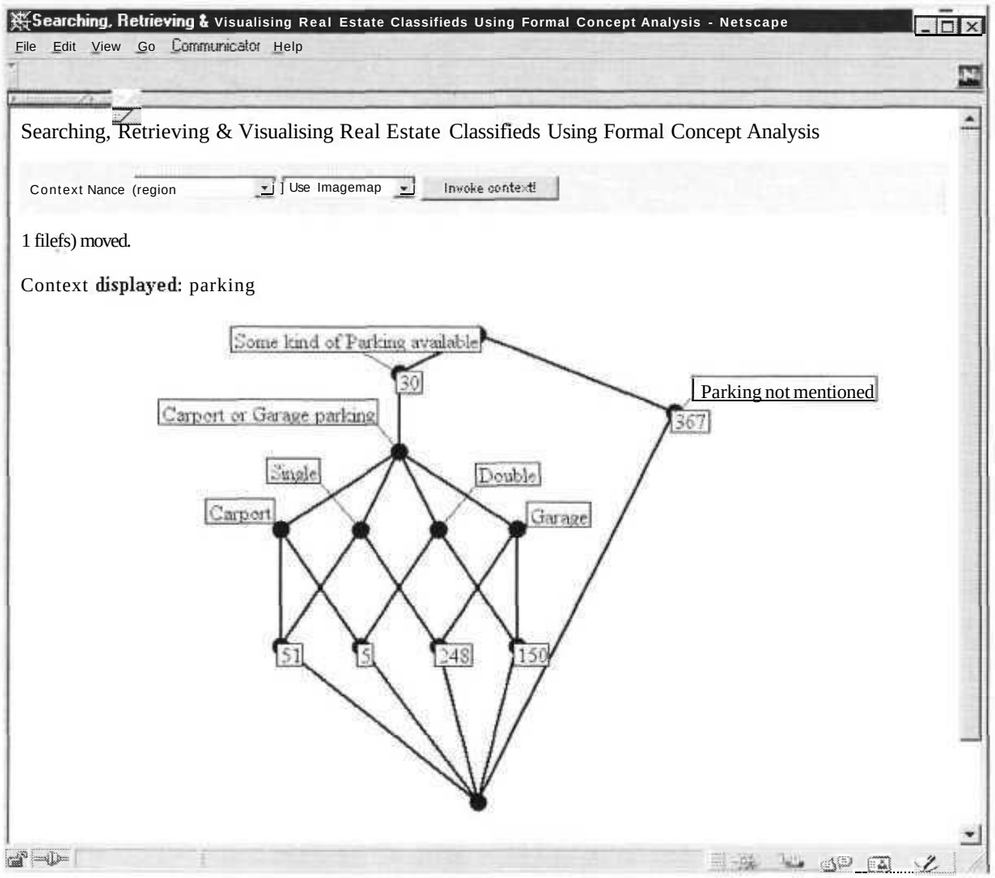


Figure 4: A concept lattices showing the types of parking available in the real estate advertisements.

at different levels of detail either focusing of properties according to some specific criteria or taking a global perspective. Although we haven't implemented zooming and nesting in the web based interface RFCA we have employed it our other systems [12, 4] we've developed. Nesting and zooming was first implemented in a system called TOSCANA[16].

By it's very nature the view of data represented by a concept lattice makes evident different levels of detail. The top concept (concepts are represented by circles) in a concept lattice refers to all objects under consideration. In the absence of zooming this will be all rental properties. As one moves down from the top concept the concepts become more specific. They refer to small and smaller sets of rental properties circumscribed by large and larger sets of attributes. For example in Figure 5 as the user moves directly downwards within the large circle one encounters the attributes for the suburb of the rental properties. Further down still following the line between large circle one encounters attributes related to parking which further restrict the rental properties under consideration.

As an information retrieval tool, zooming and nesting of conceptual scales have not been benchmarked against established IR. techniques, but the browsing metaphor is nonetheless a compelling advantage to their use.

4 Results

On the test set of 2 months worth of rental classifieds (for the Gold Coast only) extracted from a NewCorp Web site[1] (8,456 classified adverts) the system achieved logical recognition of 89% of properties, thus 11% of the property description text was discarded because price, contact number of location could not be resolved by the parser. More

[1] http://www.newsclassifieds.com.au

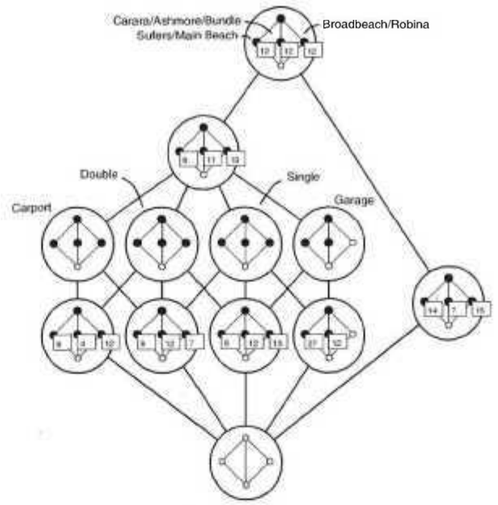


Figure 5: Example of nesting

than 1/3 of the error cases had genuine missing unresolved property listings (commonly no price was given for instance, in such a case we consider that the property listing is useless).

A LL(1) parser was constructed by hand to translate the remaining 89%, of classified advertisements into the multi-valued context representation. The multi-valued context had 64 attributes. 53 of which were single valued (true or false) attributes. The remaining 11 attributes. such as Price, were multi-valued. To access the accuracy of this translation process, precision and recall were measured for each attribute and then aggregated. A summary of the most common and most important attributes is given in Table 1.

	Location	Price	Bedroom
Frequency	100%	100%	100%
Precision	100%	100%,	100%
Recall	94.3%	100%,	98.1%
	Furnished	Car Park	Other
Frequency	26.4%	50.9%,	88.7%
Precision	100%	100%	100%
Recall	71.4%	96.3%	68.1%

Table 1: Recall and Precision for 53 unseen real estate adverts.

The precision and recall of the multi-valued attributes are calculated as the number of correctly identified attributes values as a proportion of the number of identified attributes values, and the number of correctly identified attributes values as a proportion of the number of attribute values respectively. Averaging the most important attributes — Location, Price, Bedroom, Furnished, and Car parking - weighted by their frequency yields a precision of 100% and a recall of 95% while the inclusion of other attribute reduces the recall to 90%. All real estate advertisements leave out some information about the property they are advertising because of the cost of advertising space. As a result we would expect the recall of actual information about the property being advertised to be much less. One of the strengths of formal concept analysis is that it allows the user to compose views of the data that separate objects at different levels of detail. So for example the user may have a course distinction based on cost, but a very fine distinction based on proximity to facilities contained within a single view. Table 1 shows poor recall for attributes in the group Other. When combined with the knowledge that the adverts contain only partial descriptions of the data this places a practical limit on the fineness of detail that can be usefully explored by the user. This limit would be extended if the initial data source was a database containing more extensive information about the properties for sale.

The parser being a hand crafted LL(1) parse was very fast, building the relational database storing the multi-valued context in under 8 seconds on a Pentium-Ill 300 MHz.

5 Conclusion

The purpose of this paper has been to report on a practical application of information filtering and browsing based on formal concept analysis. The system described (RFCA) is useful because it contains components that extract unstructured text from Web-accessible databases, clean the data and then perform object and attribute identification using a JavaCC parser.

Our emphasis is on the visual outcomes that can be used for text data mining showing that the visual complexity of the lattice representation can be used to explore a document collection in an intuitive human-centered interface. Future directions for the work include comparing the use of the hand crafted parser with both text classifiers based on machine learning algorithms, and the use of metadata.

References

[1] C. Carpineto, and Romano, G., A lattice conceptual clustering system and it application to browsing retrieval, *Machine Learning*, Vol. 24, pages 95-122, Kluwer Academic Publishers, The Netherlands".

[2] R. Cole, P. Eklund: Scalability in Formal Concept Analysis: A Case Study using Medical Texts. *Computational Intelligence*, Vol. 15, No. 1, pp. 11-27, 1999.

[3] R. Cole, P. Eklund: Analyzing an Email Collection using Formal Concept Analysis. *Proceedings of the European Conf. on Knowledge and Data Discovery,* pp. 309-315, LNAI 1704, Springer, Prague. 1999.

[4] R. Cole, G. Stumme: CEM - An Email Analysis Tool. *Proceedings of the 8th International Conf. on Conceptual Structures,* pp. 309-315, LNAI 1704, Springer, Darmstadt. 2000.

[5] R. Cole, P. Eklund and G. Stumme: CEM - Visualization and Discovery in Email, *Proceedings of the European Conf. on Knowledge and Data Discovery,* pp. 309-315, LNAI 1704, Springer, Prague, 1999.

[C] R. Cole, Using Force Directed Placement and Genetic Algorithms for Concet Lattice Layout, *Proceedings of Australian Computer Science Communications.* Los Alamitos, CA, 2000. IEEE Press. 2000.

[7] Michael K. Coleman and D. Stott Parker. AGLO - Publications and Implementation. *Software - Practice and Experience,* pages 1415-1438, December 1996.

[8] R. Cole, P. W. Eklund, D. Walker: Using Conceptual Scaling in Formal Concept Analysis for Knowledge and Data Discovery in Medical Texts, *Proceedings of the. Second Pacific Asian Conference on Knowledge Discovery and Data Mining,* pp. 378-379, World Scientific, 1998.

[9] B. Ganter, R. Wille: *Formal Concept Analysis: Mathematical Foundations.* Springer, Heidelberg 1999 (Translation of: Formale Begriffsanalyse: Mathematische Grundlagen. Springer, Heidelberg 1996)

[10] R. Godin, Gecsei, J. and Pichet, C: Design of a Browsing Interface for Information Retrieval, *SIG-IR,* pages 246-267, 1987.

[11] R. Godin. and Missaoui, R. and Alaoui, H. Incremental Concept Formation Algorithms based on Galois (Concept) Lattices. Computational Intelligence, Vol. 11, number 2, pp. 246-267, 1995.

[12] G. Stumme: Hierarchies of Conceptual Scales. *Proc. Workshop on Knowledge Acquisition, Modeling and Management.* Banff, 16.-22. October 1999

[13] R. Wille: Restructuring lattice theory: an approach based on hierarchies of concepts. In: I. Rival (ed.): *Ordered sets.* Reidel, Dordrecht-Boston 1982, 445-470

[14] R. Wille. Line diagrams of hierarchical concept systems. *International. Classification,* 11:77-86, 1984.

[15] P. Wille: Conceptual Graphs and Formal Concept Analysis. In: *The 4th International Conference on Conceptual Structures.* LNAI 1257, pages 2-18, Springer Verlag, 1997.

[16] F. Vogt, C. Wachter, R. Wille: Data Analysis based on a conceptual file. In: *Classification, data analysis and knowledge organsiation.* pages 131-140, 1991.

Implementing Shared Document Preparation with Lightweight Editing

Michael J Rees

School of Information Technology
Bond University
Qld 4229, Australia

mrees@bond.edu.au

Abstract

Virtually all web pages are read-only, yet the first web browser allowed users to read and edit every page. Special ad-hoc mechanisms are needed to make all or part of a page editable by a user. This paper describes Pardalote lightweight editing, a document management feature for allowing many users to share the editing of a web page using only a web browser. A brief oven'iew of how Pardalote is implemented is followed by examples of shared document preparation using Pardalote. The benefits of such web document management are discussed. Future Pardalote extensions using XML precede the closing remarks.

Keywords Shared document management, cooperative document preparation, lightweight editing, I-grains, fraglets, user interface design, computer supported cooperative work.

1. Introduction

Tim Berners-Lee describes the development of the World-Wide Web and the first browsers and servers in [1]. In 1990 the excellent design of the NextStep operating systems running on the Next machine made it very straightforward to allow any web page displayed by the web browser to be edited in-situ. Only when the pressure mounted to provide browsers on several other hardware platforms was the in-browser web page editing feature abandoned. Those second stage browsers simply displayed web pages and set the browser model that still holds today.

Although the original Mosaic browser thought to provide us with annotation capability, it was not until 1995 that MIT organised a meeting [2] to discuss methods for making web pages into a collaborative medium. Many large software companies and large research projects were represented there. Each paper presented a different mechanism to achieve collaboration via the Web. Several of these solutions are still available today as commercial products.

In the same year, 1995, the working group that was to produce the WebDAV interoperability specification, World Wide Web Distributed Authoring and Versioning [3], was formed. WebDAV [4] is an extension to the HTTP protocol that supports web document metadata, namespace management (like file system directories), overwrite protection, and versioning management. In effect, WebDAV allows web documents (any file that has a URL) to be edited asynchronously by many users, providing collaborative authoring. Of course, WebDAV support must be built into browsers and servers, a not inconsiderable implementation effort. Once achieved, however, this will provide access to genuinely collaborative web documents at last.

In the view of the author at the time of writing, WebDAV offers considerable benefit in the longer term but in the context of this paper, at the cost of a heavyweight solution in terms of software implementation. Users wishing to collaborate via web pages using WebDAV will need to wait for its use to become widespread.

A lightweight solution to collaborative web document editing is one that can use existing browsers and servers. To be truly lightweight, the solution must be a convenient one for all types of users involved:

- Members of collaborative teams who wish to jointly prepare and share web documents, the ultimate end-users
- Original authors of the collaborative web documents
- Web site administrators who install the software that makes collaborative editing of web documents possible

Probably the best example of such lightweight editing is the work of the Sparrow Project [5] from Xerox PARC. The author in [6] and [7] has described examples of Sparrow's capability. The first user type, end-users, is very well catered for in Sparrow. The user is presented with a very simple and intuitive user interface to edit nominated sections of the web document (HTML page). Edits are restricted to the textual content in specially marked locations.

Although the Sparrow user interface is very simple, CGI scripts are used to update the web document contents as editing proceeds. This entails a round trip to the web server for every update, and is thus inevitably slow in response over congested Internet links. Sparrow document authors (type 2 users), who wish to incorporate editable sections in their pages, must learn and use special tags, which are interpreted by the Sparrow CGI scripts.

The author of this paper determined to adopt the simplicity of the Sparrow user interface and to overcome the response time problems. This led to the Pardalote project using a new approach to implementing lightweight in-situ editing. At the same time, Pardalote addressed the issues of web document authoring to make it extremely straightforward to incorporate lightweight editing in specified parts of the document.

WebDAV and Pardalote represent two end of a spectrum of in-situ editing of web documents, from heavyweight to lightweight. Before describing Pardalote in detail from Section 3 onwards, some middleweight solutions are discussed in the next section.

2. Other middleweight approaches

In the middleweight-editing category, an impressive Australian solution called EditLive [8] is worth mentioning. End-users are presented with web documents with editable sections. A reasonably rich DHTML editing tool is provided, so that user can make sophisticated, formatted edits of web document content to which they are given access.

In the context of this paper. EditLive is a middleweight solution because it requires end-users to download and install the editing software. While installation is very straightforward, this additional install step can lead to problems for some users.

EditLive software on the client exists in the form of a browser plug-in specifically implemented for a particular browser. On the Microsoft Internet Explorer (IE) platform, EditLive makes full use of the DHTML editing component. This provides the user access to sophisticated, WYSIWYG editing with control of features like:

- Fonts, font sizes, colours and weights
- The standard HTML styles like headings, paragraphs and lists
- Tables
- Image insertion and placement

The end-user need not know HTML and is presented with a word-processor-like user interface. This can be used in a very simple way, but the more advanced formatting features place the end-user editing into the middleweight category. In addition, the DHTML editing control does not possess all the usual editing conventions, seemingly lacking the simple use of the Backspace and Delete keys to delete characters adjacent to the cursor.

Web document authors have some work to do. While simple web page templates can be provided, authors need to understand scripting to design their own templates. The scripts call upon the client-side EditLive plug-in objects to activate the editing components. While not difficult, scripting is beyond many web page authors.

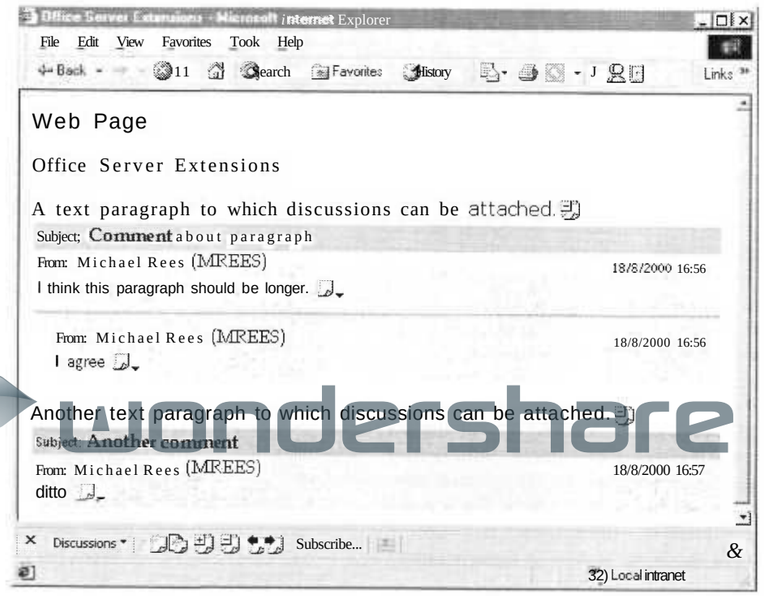


Figure 1. Microsoft Office Server Extensions with in-document discussions.

EditLive uses File Transfer Protocol to send the new edited information back to the web server in order to update the file. Web page authors and web site administrators are provided will settings and tools to control which end-users have this access and for which files. Again, this requires that these users have suitable knowledge and training in order to utilise the required access control.

Mention should be made in this section of a couple of in-situ annotation solutions that effectively provide the end-user with a simple editing facilties within web pages. Microsoft Office 2000 provides the Office Server Extensions (OSE) [10], which run with the Microsoft IIS web server. When using IE to view a web page resident on IIS with OSE installed, the end-user selects the online discussions tool built into IE. This then presents a view of in-document discussions as shown in Figure 1.

Using special file and folder icons the OSE discussion boxes are attached to the paragraph of text to which the comments refer. Each comment is tagged with the author, date and time. Other users may reply to comments, and edit and delete their own comments.

While the browser gives the impression the discussions are integrated into the web page, the discussion text is actually stored in a SQLServer database on a machine on which the OSE are installed. The discussions database can be stored on any machine anywhere on the Internet. When IE prints the web page, the discussions are interleaved with the contents of the page. Another feature of OSE not relevant here is the ability for users to be informed by email of any discussion changes made by other users.

Once again. OSE presents end-users with a lightweight editing task-only text is entered and modified. Web document authors have the easiest task of all; they simply arrange their documents to be part of a web site that grants end-users access to OSE discussions. Web site administrators have to install OSE and manage user accounts in order to grant access to the discussions facility, and arrange for routing email notifications.

Another excellent Australian software development in the area of middleweight editing is the PageSeeder package [9] from Weborganic Pty Ltd. What is now PageSeeder grew out of the need to allow remote learners the opportunity to enter discussions into shared web pages as part of their online education materials. For simplicity. PageSeeder originally allowed users to read in-page discussions entered by other users, and enter their own discussion points by submitting email requests.

In its latest commercial fonn, PageSeeder allows end-users to click on the proprietary seed symbol, , and enter their comments/questions/answers at that point in the document. A new browser window containing a suitable form pops up for this entry process.

For each seeded entry, PageSeeder add an additional line into the web page at the end of the seeded paragraph. The line contains the subject of the entry as a hyperlink together with author and date information. Clicking on the hyperlink pops up another window containing the full text of the entry together with menus for creating new entries and replying to the current entry, and so on. This view is shown in Figure 2.

Each user interaction with PageSeeder requires a round trip to the web server, and is implemented using Java applets embedded in the web page. These applets operate in both the major browsers and control which end-user can view and/or edit the seeded comments. Web page authors must learn the appropriate additional HTML tags to insert the applet objects and set their parameters. PageSeeder provides web page uploading and downloading for those users with page

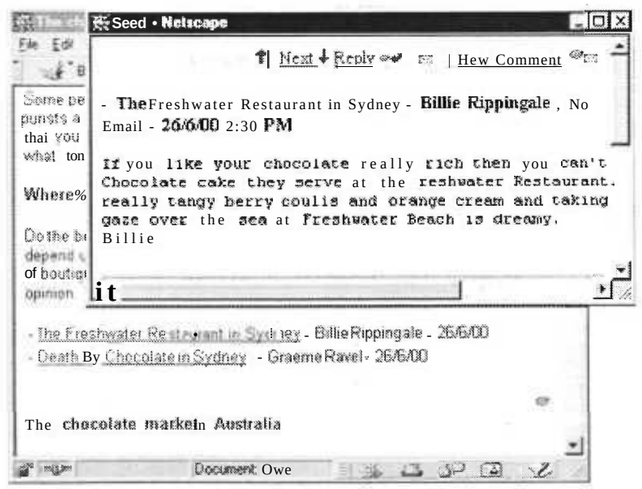


Figure 2. PageSeeder seeds and comment expansion.

authoring permission. This makes page authoring a middleweight solution, as is the web site administration. Details of the latter will soon be available at the Weborganic web site.

3. Lightweight editing with Pardalote

Pardalote has been expressly designed to offer a lightweight solution to all three categories of users: end-users, page authors and web site administrators. This sets it apart from all in-situ editors mentioned so far. Version 4 of Pardalote described in this paper achieves this aim admirably. Being an experimental system, however, this aim is achieved at the cost of making Pardalote browser-specific–in this case Microsoft Internet Explorer only.

Pardalote 3 is described in [7] and incorporates these ultra-lightweight features:

- All in-situ web page edits are achieved with instant response times without requiring a web server round trip-an extremely simple and responsive user interface for end-users
- After linking to a CSS style sheet and adding one <script> tag, editable sections of web pages are simply marked up using style sheet classes-page author training is minimal
- One simple ASP web page containing less than 10 lines of script is needed to update edited web pages as part of a small FrontPage web. installed once-web site administration is minimal

This third version of Pardalote uses Microsoft DHTML behaviors, a proprietary feature of Cascading Style Sheets. This slowed the downloading of pages containing large numbers of Pardalote editable sections. In addition, an edited Pardalote 3 page adds additional, hidden HTML tags that make subsequent editing with commercial editors like FrontPage more difficult. (A similar problem is exhibited by all middleweight solutions mentioned above.)

Pardalote version 4, described in this paper, has been completely reimplemented avoiding DHTML behaviors, and using asynchronous update for no-wait saving of edited pages. Also, no hidden HTML tags are used, so that once pages have been edited with Pardalote, they remain easily editable by editors like FrontPage and others.

For comparison, Figure 3 shows the user interface for editing paragraphs. Pardalote uses the ✍ symbol to indicate which parts of a page can be edited by the end-user. For improved consistency in version 4, all parts of the page controlled by Pardalote have a light blue (grey in black and white) background.

In Figure 3 the user has just clicked on the text of the first bullet point to bring up the edit panel (this occurs instantly since all the HTML is generated within the browser scripts). The text of the list item can be edited within the text box form element. Clicking on the Save button changes the page to reflect the new contents, closes the edit panel, and returns the page to its orginal format.

Each part of the page marked with ✍, is called an I-grain (short for infonnation grain). I-grains can be any text tag such as headings, paragraphs, list items, table cells, addresses, and so on. Every I-grain can be edited, and there can be any number of I-grains on each page. This allows great flexibility in the layout of the page for shared editing. Page authors can impose rigid stmcture so the page remains readable and consistent with other pages in a web site. On the other hand, very little non-editable text may be used to

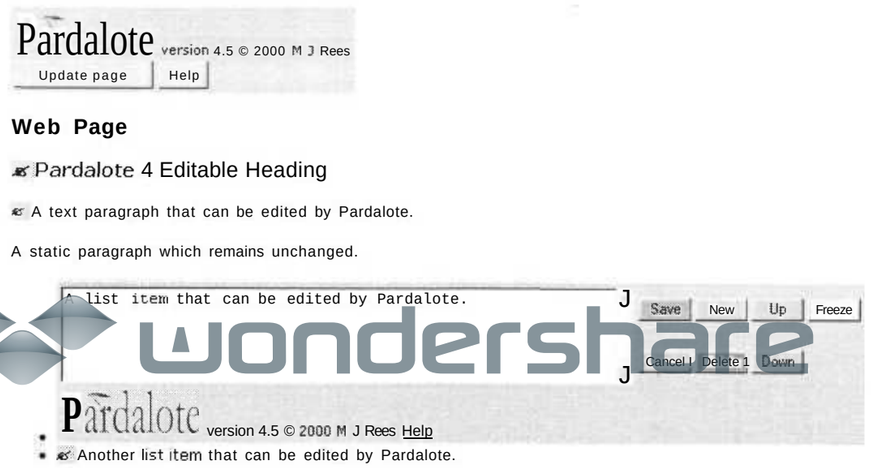


Figure 3. Pardalote 4 in-situ editing.

allow the group sharing the editing to make significant changed to page layout as modifications are made.

Note the range of editing operations:

- New I-grains can be created from any existing I-grain. The new I-grain is inserted after the current I-grain.
- I-grains can be moved up and down in the page any number of times.
- An I-grain can be permanently deleted from the page.
- An I-grain can be frozen. In other words, it ceases to be an I-grain and becomes text that is static and can no longer be edited by Pardalote.

Pardalote 4 web page authors need only add the following two lines to the <HEAD> tag of a web page to switch on editing capability:

```
1. <LINK id=lweStyle
   href="/lwe/lweStyles.css"
   type=text/css rel=stylesheet>
2. <SCRIPT language=JavaScript
   id=lweScriptInclude
   src="/lwe/lwe.js"></SCRIPT>
```

Previously, the web site administrator will have created the /lwe FrontPage web that contains:

- Pardalote style sheet, `lweStyles.css`
- Pardalote lavascript page, lwe . j s
- Image files and help pages

In total, the Pardalote files occupy a lightweight 46 Kbytes of disk space. Of course, the web site manager must set write permission for the default web server user account in all those folders holding Pardalote pages. This allows the Pardalote page files to be updated by the ASP scirpt.

Once the style sheet is linked to the page, the FrontPage editor shows the class in the style list box. Thus to make a piece of text editable, the user just selects the appropriate style. For the list item I-grain in Figure 3 the Pardalote style class is lwePara-f defined in the style sheet.

Style classes are provided that provide the page author with different combinations of the above editing operations. For example, if the end-user is intended only to modify an I-grain, then the page author applies style class lwePara-e. About a dozen combination style classes are supported for paragraphs.

All Pardalote page edits are temporarily stored in memory of the end-user's browser. To make the changes permanent and visible by other users, the Update page button in the control panel inserted at the top of each page (and at the bottom) must be clicked. This sends the complete HTML of the page, encapsulated as XML, to the web server, where an ASP script overwrites the original page. The update operation is implemented asynchrounously, and the user can continue to edit the page while the update takes place.

Pardalote is designed for small collaboration teams where all users know and trust each other. All users sharing a web page have equal privilege to insert, change, move and delete any I-grain. A simple check is made in the ASP script to detect whether another user has updated the shared page while the current user has been making changes I-grains. If this happens, the update does not take place, and a warning message generated. For typical small

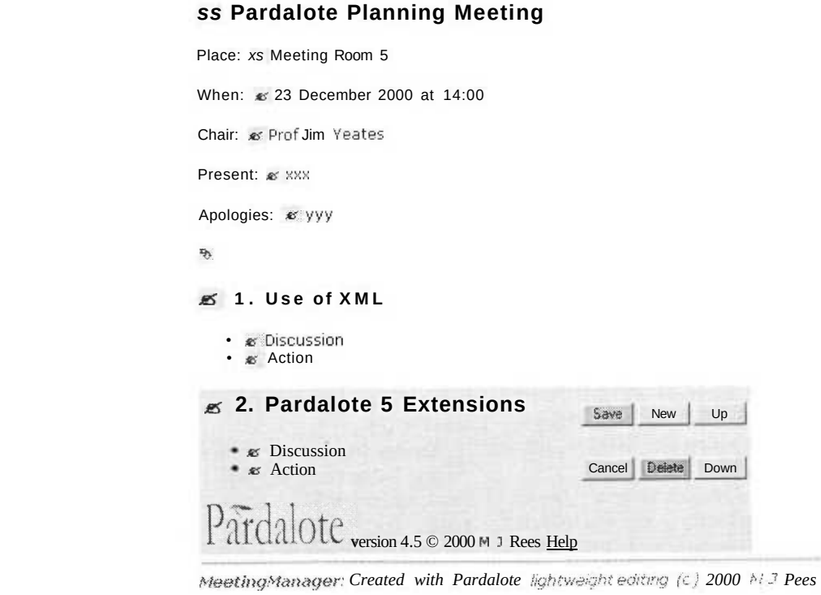


Figure 4. Pardalote meeting document management.

changes, the user can copy changed text to the clipboard, refresh the page to view the new version, redo the edits, then update again. Such a situation is extremely rare, and the solution fits well with the lightweight philosophy built into Pardalote.

4. Document management with Pardalote

A simple meeting manager document constructed as an editable page is used to demonstrate the document management capabilities of Pardalote. This example allows a single web page to support the following phases of holding a meeting:

1. Announce meeting: title, date, time and place
2. Agenda setting: initial and refinements
3. Taking minutes during meeting, and formulating action items
4. Minutes update following meeting

This example makes use of a composite editing structure supported by Pardalote called the I-grain pod. A page author constructs a pod by creating a <D1V> tag holding one of more I-grains interleaved with static text, if required. A pod style class is then assigned to the <DIV> tag.

Figure 4 shows the upper part of a meeting document page soon after it has been created. Note that the I-grain pods are represented by the pod edit character. ✎. The user has clicked on the pod edit character of the second I-grain pod to display the pod edit panel shown near the bottom of Figure 4. Note the same edit functionality buttons are available allowing pods to be deleted, moved and created. However, no editing of I-grain pod content is possible in this view.

To edit individual I-grains within a pod, the user clicks on the normal I-grain edit character as before. This allows I-grains within a pod to be added, deleted and modified using the same mechanisms as already described.

One other Pardalote feature is shown in Figure 4. The numbers prepended to the agenda item titles are generated by the Pardalote script by inserting the <SPAN> tag:

```
<span class=pardNumber></span>
```

The SPAN will be replaced by the next number in sequence. Moving agenda item I-grain pods relative to one another will cause all to be renumbered automatically.

When the meeting document is first created, the users likely to attend are informed by email. Accessing the Pardalote page, they are able to edit existing agenda items and add new ones. Meeting organisers can amend the meeting title, location, place and chair at any time.

During the meeting, the minutes secretary can use a machine (perhaps a laptop with a wireless modem) to edit the minutes. Projecting the screen of this laptop allows the whole meeting to view the minutes as they are created.

Following the meeting, users can continue to refine the minutes. Eventually, the document can be frozen to prevent further changes, as described below. In this example notice that only a single meeting document is ever produced. There is thus only one version in one place that supports all phases of the meeting.

Shared documents will normally have a final version where no further editing is needed. The Pardalote editing features and special editing characters will no longer be needed. In effect, the editing is frozen.

Pardalote provides such a feature by adding an extra attribute to the <SCRIPT> tag inserted in the document head. Adding:

```
lweFreeze=""
```

will cause the Pardalote document control panel to be displayed as shown in Figure 5. When the user clicks on the Terminate editing button, all trace of Pardalote editing is switched off. and the page becomes regular static HTML.

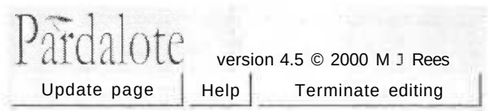


Figure 5. Button to freeze editing.

If the page author adds the attribute:

```
lweFreeze="../readonly/dual.htm"
```

then the document control panel is displayed as shown in Figure 6. In this case, when the user clicks on the hyperlink "../readonly/dual.htm" Pardalote will both update the page to allow continued editing, and save a second, frozen copy of the page in the indicated page file.

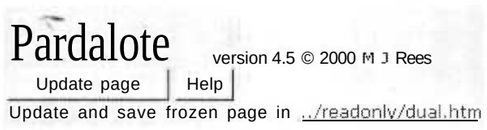


Figure 6. Update and save frozen page.

Such a facility is useful when the collaboration team wish to share a read-only copy of their latest document with a wider audience.

Only the Pardalote paragraph and pod I-grains have been described in this paper. Pardalote also supports section, hyperlink and in/out indicator I-grains. Examples of these additional I-grains can be found in [7].

Section I-grains are particularly useful for lists of various kinds such as project lists, comments, news items, meeting notices, and so on. Available with all I-grains, the Pardalote extend feature is especially helpful for web pages containing lists.

All list I-grains except the last in the list are given style classes that limit users to modifying the text

contents (in other words, no delete or move). The last I-grain in the list is given a style class with extend capability, such as lwePara-x. When the New button (see Figure 4) is clicked, the **current** I-grain loses extend capability, and the newly created, last I-grain retains it. This allows users to extend the list continually but all users are prevented from deleting or moving any list I-grains. This provides simple document management of pages containing shared lists.

5. Conclusions and Future Work

Pardalote 4 provides a genuine lightweight implementation for in-situ editing within a web page. End-users have no need to download any special software, they simply use the Internet Explorer 5 browser. Page authors insert two lines of HTML into their pages, and then use their preferred web page editor to apply Cascading Style Sheet classes. Web site administrators copy one, exceedingly small web site (lwe), and set write permission on selected folders.

A comparison of Pardalote with other solutions mentioned in this paper is presented in Table 1.

Table 1. In-situ page editing solutions compared.

Solution	**End-**users	Page Authors	Web **Site** Administrators
Pardalote	LW	LW	LW
Sparrow	LW	MW	MW
EditLive	MW	MW	MW
OSE	LW	MW	MW
PageSeeder	LW	MW	MW

Key: LW = lightweight. MW = middleweight

Pardalote, of course, simply addresses in-situ editing, and does not provide any web site management facilities that are present in many of the competing solutions. Nonetheless, Pardalote is highly effective for single-web-page document management controlled by small collaborative teams.

Development and evolution of Pardalote is continuing. Pardalote 5 is being implemented, heavily based upon XML and XSLT. Each I-grain is represented as an XML data island. XSLT style sheets are applied dynamically when switching between the display form and editing form of an I-grain. With this approach it will be possible to save each I-grain individually as it is changed, thereby considerably lessening the clashing update problem.

At the same time, each I-grain represents a fragment of information to be shared, a fraglet. Since these fraglets are represented in XML, they may be identified and stored separately in an XML-oriented database. Once in this representation, fraglets can be searched, filtered and otherwise managed. In addition, they can be passed for processing to other, related applications for summarising. merging and reformatting into other types of information. Processing and manipulation of fraglets will fonn a large part in Pardalote 5 development, and will be presented in future papers.

References

[1] Berners-Lee, T.. Weaving the Web. 1999, London: Orion Business Books.

[2] MIT Workshop on WWW and Collaboration, http://www.w3.org/CollaborationAyorkshop/.

[3] IETF WEBDAV Working Group, http://www.ics.uci.edu/pub/ietf/webdav/.

[4] Wiggins. J.W.a.M.. WEBDAV: IETF Standard for Collaborative Authoring on the Web. IEEE Internet Computing. 1998. September - October: p. 34-40.

[5] Chang. B.-W.. In-Place Editing of Web Pages: Sparrow Community-Shared Documents. http://www7.conf.au/programme/fullpapers/1929/com 1929.htm.

[6] Rees. M. User Interface for Lightweight In-line Editing of Web Pages. in 1 st Australasian User Interface Conference 2000. Canberra: IEEE. February: p. 88-94.

[7] Rees. M. Implementing Responsive Lightweight In-page Editing. In AusWeb 2000. Cairns: Southern Cross University Press. June: p. 265-283.

[8] EditLive. Ephox Pty Ltd, http://www.ephox.com/.

[9] Weborganic Pty Ltd. PageSeeder, http://www.weborganic.com/.

[10] Microsoft Office Server Extensions, http://www.microsoft.com/Office/deployment/chap13.htm

An Experiment in Light Workflow

Stewart Baillie, Anthony Bennett, Anne-Marie Vercoustre, Ross Wilkinson

Division of Mathematical and Information Science
CSIRO
723 Swanston St, Carlton 3053, Australia

[Stewart.Baillie, Anthony.Bennett, Anne-Marie.Vercoustre, Ross Wilkinson]@cmis.csiro.au

Abstract

Workflow tools have been successfully applied to automate work in many situations where the work is well regulated, there is a stable pattern of work, and there is a sufficiently high volume or sufficiently high importance to justify the cost of automating the activities. In many other circumstances there is a very mixed story of success and failure of workflow implementation. The Web also has changed work practices and increased the role of electronic documents, in particular, forms, as a support for many distributed tasks. In this paper we explore using a workflow approach based on fully self descriptive documents, that embed the information and instructions necessary to support processing the document, within the document. The traditional workflow engine or server that is typical of current workflow tools is discarded, but the document still allows a full work process to be applied, without necessarily enforcing the process. Ideally one would need a Web browser, and an email client, and no workflow system at all. This paper shows how this is not quite possible, but one can build a very small supporting application to achieve light workflow.

Keywords Workflow, Document Flow. Web-based Workflow, XML, XSLT, Co-operative work.

1 Approaches to Workflow

There are various ways companies implement their internal procedures to effectively accomplish their routine tasks. These range from manual approaches to fully automated, and can involve one person or several.

In a manual approach, a task is modelled or at least described in a procedural manual. Workers take responsibility to carry out the steps described in the manual, and use standard office tools - word processors, email and web browsers - to carry out the work. Often this works well, but it relies on human validation and execution, which is prone to error.

The aim of a workflow system is to automate in some part critical business processes within an organization. According to the Workflow Management Coalition, workflow is:

'The automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules'[1].

There are numerous workflow products on the market, some of them now quite mature [4], [8], [9], [10]. The approaches taken by these products can be placed in one of three classes [11]:

- E-mail or Message based. These systems only require a pre-existing e-mail system to operate. Generally, tasks are mailed to the next person in the process, along with any relevant documents for them to complete the task. Once their stage is complete, the information is mailed back to the server.
- Web Based. These systems make use of a web server, and a browser to perform the workflow This gives the advantage that users may access the workflow system from essentially anywhere. The main problem with the system is that client's must log in to receive the list of tasks that need to be completed.
- Monolithic and proprietary based (production based). Applications in this category are generally quite large. Users access the system through customised client software, which attaches through custom communication techniques (ie not e-mail or http protocol) to a server. These systems attempt to cover every scope of current workflow technology.

[1] www.aiim.org/wfmc/standards/docs/glossy3.pdf

Common to all products is a server for controlling the workflow and a commercial database system on the server side for the storing of information. They also all require substantial process modelling, then a workflow description to support that model.

In order to exploit the potential of the Web. many companies now provide a Web interface to their Workflow Management System and a notification through e-mails, e.g. [8], [9], which results in systems that may cross over the boundaries of the classes defined above.

In addition to workflow approaches it is also worth mentioning the cooperative work approach, where a group work together to cany out an information task, using tools to make relevant information available, as well as tools that help the group cooperate. This work might still require workers to use a manual, but may better support work in an ad hoc environment when the process is not clear, even though the outcome may be. The cooperative approach tends also to move towards a Web-centred approach [1], [3]. The cooperative approach may benefit the introduction of light workflow in order to improve the coordination of work carried in these environments.

In this paper we will concentrate on support for those information tasks that can be earned out by the production and manipulation of documents. Thus we concentrate on document flow.

2 The Need for Light Workflow

Commercial workflow tools are based around a complex, and often sizeable server. This server acts as the hub for executing processes, including the storing of process data, view generation and the enforcing of rules, by data validation, and forwarding.

This approach works very well for workflow types that are frequently executed and static in approach. However, as workflow types become less frequently executed, an increased requirement for ad-hoc processing arises, or a need to cross company boundaries appears, systems taking this approach become less effective.

Effectiveness can be compromised for a number of reasons, including cost, human behaviour and incompatibilities with other workflow products.

In order to support more distributed workflows, and those less well catered for by current tools, a system needs to be smaller (and thus more cost-effective), cross platform compatible, and be simple enough to use so that ad-hoc processing is actually effective.

A typical example is the Conference paper review process which is often earned out via a central server where reviewers enter their review through a form. The drawback is that reviewers are not always connected to Internet when making their review and may also wish to keep a copy of their review for future use or in case the review is lost. They would rather connect to the server to get the fonn on their computer, complete the form using their local browser, save it locally, and send it when they are later connected, by opening it again and clicking on a submit button.

For this we propose a system of light workflow, where much of the server functionality is removed, such as the enforcing of flow rules, and is implemented in a self descriptive 'transaction document'. This document moves from client to client, providing everything that is needed to process the document on the client side.

In order to develop the idea of light workflow, we will first present an example application appropriate for this approach, and our system that implements this example.

3 An Example Application - The Paper Submission Process

In order to test the concept of light workflow, a test application was needed. Chosen was the CSIRO process for getting a paper approved for submission to publication. This becomes a 'workflow type' (often refened to as a process or business process). Broadly speaking, the paper submission process (if executed in correct order) consists of the following steps:

- Gain approval to write the paper from project leader (providing paper title, abstract, etc).
- Write paper
- Obtain reviews of the paper.
- Make changes according to the reviews, and submit paper for acceptance.
- Determine what to do if paper not accepted.
- Obtain copyright agreement and modify according to CSIRO guidelines.
- Make changes to the final paper and submit.
- Send copy of final paper to publications officer, and record the publication on the publications database.

Why do this electronically? Often, many steps of the process are left out, especially those that are of no direct consequence to the author(s). This leaves scattered records of papers that have been accepted for publication. By doing this process electronically, we may make automatic records of accepted publications, and provide a record of the transactions that took place to get the paper published, should this ever be needed.

The other reason is in tenns of light workflow, this process consists of some challenging points to consider:

- Some steps are simply document uploads. These need to be supported in an attractive manner, so that users will still want to upload these documents (to form a permanent record). Users need to see the value of recording such documents.

- The process requires flexibility. Firstly, for some steps, users complete them in multiple ways. This is typical of reviews. The system must make it easy as possible for users to complete such steps in the manner they are used to. Secondly, ad-hoc processing is required - not all steps will necessarily be completed, and may also be completed out of order. This needs to be supported.

In order to use this as a test platform, we created a detailed step-by-step breakdown of an appropriate process. This involved discussions with researchers (users) to obtain a model of how they approach the process of writing a paper, and what would be important to them in a system. This resulted in a detailed step-by-step breakdown of the process.

Using this, work could begin on the actual light workflow system.

4 A Light Workflow System

The light workflow approach provides a system that is document-centric, as opposed to the data driven approach of most commercial tools. This technique aims to provide a portable, workflow enabled document. The design of this approach keeps in mind the following points:

- The system should be document-centric (self-descriptive).
- Clean abstraction of document data, views and flow.
- Web standards should be used, to allow the workflow to function on any client with a modern browser. (XML, Javascript, Mail).
- Ad-hoc/flexible processing is required.
- Processing is moved from the server to the client, to remove the need for a server.
- User of standard data model and public API for data processing (XML/DOM, XSLT).

Instead of a server acting as the hub of activities, the 'transaction document' becomes the central component. The transaction document stores both the data and the instructions that describe the processing of each step, such as data validation and storing, and the relationship between steps.

When a step is completed, the entire transaction document is forwarded to the next person in the flow.

As is the characteristic of workflow applications, the transaction document produces a differing view of itself for each different step. Additionally, processing instructions may or may not check which step was executed previously (and thus check if it is their turn or not) thereby allowing for ad-hoc processing, by being able to complete any step at any time. To utilise this ability, some form of 'step selector" is available, allowing quick navigation and examination of each of the steps available (as seen in the left frame of figure 4).

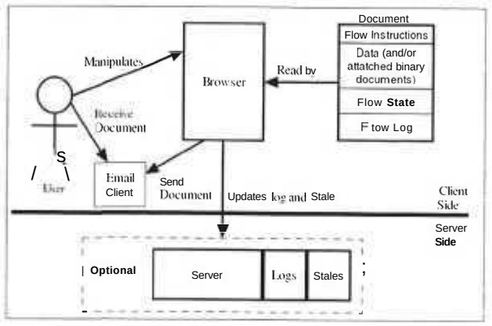


Figure 1 - A simple Architecture for a light workflow system

Figure 1 illustrates an architecture for light workflow. A browser is used to parse the transaction document. The transaction document is rendered according to the step, and the user interacts with this. Data is input to the system, and the instructions for that step process this data, modifying the transaction document as appropriate. When processing is completed, the updated document is sent to the next person in the process.

A server is still presented in this architecture, but is optional. However, often is important for a user to be able to find the current state of a process, or a need

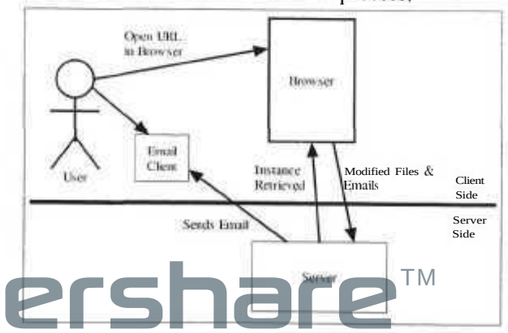


Figure 2a - Compromise made due to issues with browser security

to analyse logs of workflow usage will arise. In this case, the server acts as a 'message board' for this data.

If this functionality is not needed, then it can be left out.

However, when attempting to implement this architecture, we needed to make some concessions. Figure 2a illustrates these concessions, whilst Figure 2b illustrates the implementation architecture.

Technically, the architecture presented in 2a & 2b differs greatly to the architecture presented in figure 1. but conceptually, they are similar.

The primary reason for differences in architecture is the default security level in browsers. It is not possible to do any sort of file manipulation on the client (other than loading XML documents), nor is it possible to automatically send an e-mail. It is possible to set non-default security to circumvent these problems, but this creates additional tasks to allow functionality, and users may not be happy with this scenario, given the recent high profile of internet scripting viruses. Thus the server is now used for two additional tasks - storing the transaction documents (and related files), and sending e-mails.

The server, shown in Figure 2a, runs on top of an Apache web server, which is equipped with the Jserv module to allow it to run Servlets. Three separate items are served from this:

1. Workflow Engine - This is not technically an engine, but a collection of small utilities. Users connect to this to create a new workflow instance, to specify which instance they want opened, (and to what step) and to view the status of other instances, and log files.

2. Servlets - These allow the transaction documents to access the facilities of the server to create new instances, send e-mails and save modified instances back to their original location.

3. Instances Folder - This folder stores the templates and the running workflow instances. The template is a transaction document that represents a particular workflow type. Copies of these templates arc made each time a new workflow is created. These arc referred to as workflow instances.

```
- <paper>
  + <people>
  + <paperDetails>
  - <process>
    + <stagePresentation>
    + <logs>
    + <owner>
    + <completedStates>
    </process>
  </paper>
```

Figure 3 - Structure of the Transaction Document, from the sample workflow type.

The transaction document is constructed from two files, an XML document and an XSL document. The XML document consists of the workflow data, and the instructions for each step. The instructions currently are specified in JavaScript, for interpretation by a web browser. Figure 3 shows the structure of such a transaction document, in this case for the paper submission process.

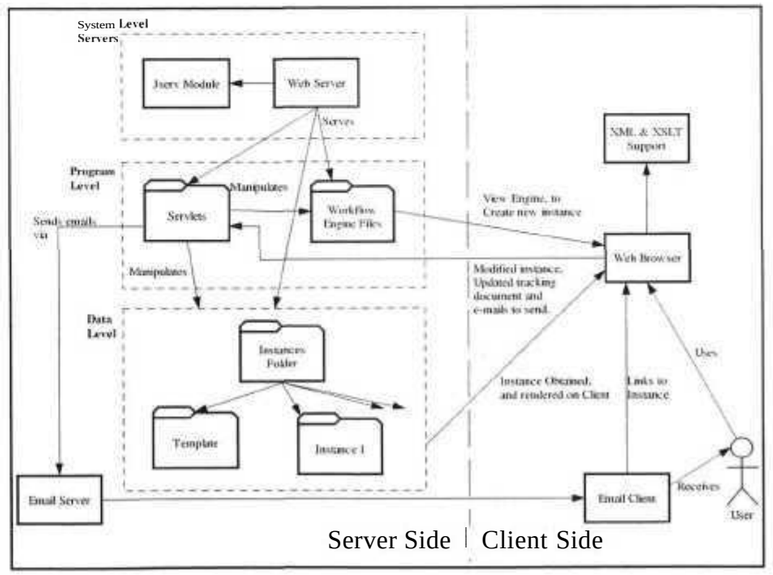


Figure 2b– The architecture of a implementation of a light workflow system.

This document is broken into two parts - the workflow data, and the transaction document data. The former is specific to the workflow type. For the paper submission process, there is an element (people) for storing data about the people involved in the process, and another for storing data about the paper itself (paper details).

The second part to the document structure is the process data. Each workflow type has this structure, completed according to the particular flow. The stagePresentation tag (under the process tag) represents all data that is needed for displaying a step, except for the forms, and the actual appearance (which is handled by the XSL document). Therefore for each step, it contains the processing instructions, text to be displayed to the user, the contents of e-mails and the explanation to be displayed to the user once processing is completed. The Log tag records each step that takes place, when and what data changed. Owner represents the creator of the process, and completedStates records the states that have completed fully.

The XSL document consists of display information. By interpreting a parameter provided to it at display time, the XSL creates a display appropriate to the step to be viewed. The appropriate text and code is extracted and formatted into the display, and then the appropriate form inserted.

Figure 4 shows an example screen of the paper submission process running. The left hand 'frame' shows the stage selector structure. Clicking on any of the stages automatically opens it on the right hand side. The right hand side has been totally rendered from the transaction document, and is displaying the contents that specify each field.

The user can also be prompted by e-mail to open an URL. When the URL is clicked on. the appropriate transaction document is opened, rendered and displayed to the client according to the step specified.

It is important to realise that despite the reliance on a server, conceptually, the two different architectures are almost the same. If you were to remove browser security, then only minor adjustments would be required for the system to work purely client side. The transaction document carries everything with it to describe the workflow.

5 Discussion

We have argued that workflow systems are valuable for some activities in a workplace, but that there are times in which a fully prescribed workflow is not possible, or a commercial workflow tool is not effective. We thus designed and created a light workflow system to address some of these needs. In order to evaluate this system, we can analyse the approach against the following criteria:

- Does the paper submission process work?

Yes. The system has been used successfully to gain pennission to write a paper, right through to publication.

- Is the effort needed to create a new application too high?

At this stage, yes. For a typical process of 8 - 10 relatively different steps, it look us several months to fully complete a new process. This will no doubt improve as expertise is gained, and generic libraries of functions are obtained. However, the development time is still likely to be too high. Most current workflow applications include a graphical tool for specifying workflows, and we should assume such a graphical tool to be built for our system. Prior to this, we need to develop a language, preferably in XML for specifying the processing details for each step. This language should include XML fonn definition and manipulation[5], variables, and calls to DOM functions

- Can a workflow be easily changed?

Freedom to change and alter the system depends largely on the level of change required, but generally it is not an easy task. For example, adding a new data field may require updating the DTD, altering several forms, and including the new data field into processing code. Identifying where changes need to be made, and then testing them, is likely to take several hours. This is again a call for a process creation tool - without it the cost of defining and maintaining a flow becomes too high.

- We haven't been able to get away with just using a browser, email clients and "smart" documents. Is the system still actually light?

Needing a server to complete basic tasks is a compromise on the original design. However, we believe this is still a light workflow system. Firstly, to its cunent stage, it has not taken very long to implement. None of the complexity that you might find in a commercial workflow tool is present. Secondly, very few changes would be needed to allow the system to run purely on the client side, and we could even do this now on a machine running Outlook. (We could easily reduce security on browsers, and if necessary just use a public e-mail server, which does not really comprise the design).

- What are the limitations of this approach?

There are a number of limitations of this approach that need to be considered.

- Without a server, this approach does not support central functionality such as stage and audit tracking for workflows. However this additional functionality could be provided by external services to which the workflows would send appropriate infonnation.
- Since we do not enforce a flow, we rely on the users to use the system conectly. We have less of a guarantee that the process will be completed, or completed conectly. A commercial workflow app enforces the rules more rigorously, and has the capacity to send out reminders, set deadlines etc. However, if we don't display a stage selector, the flow is then enforced, so enforcing of rules may become an option that can be selected when a new instance is created.
- Since the XML is processed on the client side, we can't easily prevent users tampering with data that they should not. A 'trust' system is worked on. where it is assumed that users will have no interest in completing a stage that is not theirs, or changing data that they should not. In an environment where this is not the case, we may need to extend the system to allow for user authentication, placing an increasing need on a central server, or support the use of digital signatures on sections of the document.
- Data is inherently less secure, since it may be located on multiple clients, thus making it easier for unauthorised individuals to obtain data.

6 Conclusions

We have demonstrated the feasibility of a light workflow based on a document that carries all the data and instructions necessary to its processing. Should the ability to save locally, and send e-mails without user intervention arrive, then we can have a totally client side workflow system. This is what makes the major difference of our approach with cunent Web-based Workflow systems such as Lotus Notes and others. Another advantage is the use of standard Web languages and processors, such as XML, XSLT. and Javascript that allows the transaction document to be reused and further processed by new or external applications, which is not possible with a proprietary format.

We believe that this system can be effective for workflow types that are distributed, infrequently executed, require flexibility (ad-hoc) or are driven by documents. This system has the advantage that any client with a modem browser will be able to participate in the flow. This allows company boundaries to be easily crossed, e.g. the purchase order to an occasional supplier, or where an ad-hoc group has formed, e.g. a conference review process.

However, it is unlikely that we can ever move totally away from requiring a server, unless we change the base technologies and go away from using only standard technology. If we used our own custom technique, we would need our own client side parser/viewer.

There are times where light workflow is less appropriate, and a centralised approach is best. As noted, these are times where the flow is frequent, and unchanging, or the process is data driven. Central systems also have improved security, and allow the

Figure 4 - Screen shot from running paper submission system.

generation of complex statistics on usage. This may be important for improving efficiency of processes, through identification of sections of the process that result in a delay.

Finally, there are also times where no form of workflow automation may be best. Such situations might be in collaborative work, where group-ware type systems are more appropriate, and situations where for legal or security reasons, only hardcopy is acceptable.

Further testing of the paper submission process, as well as other types of workflows will be conducted to more fully assess the capabilities of this approach.

References

[1] Natalie Glance, Antonietta Grasso, Uwe M. Borghoff, Dave Snowdon and Jutta Willamowski, Supporting Collaborative Infonnation Activities in Networked Communities, In: Bullinger, H.-J., Ziegler, J. (eds.): Proc. 8th Int'l Conf. on Human-Computer Interaction (HCI'99), Munich, Germany, August 22-27, 1999.

[2] Koch. T.: XML in practice: the groupware case, in Proceedings of IEEE WET ICE Workshop 1999, Stanford University. http://bscw.gmd.de/Papers/wetice99/wetice99.pdf

[3] Bentley, R.. Horstmann. T., Trevor. J.. The World Wide Web as enabling technology for CSCW: The case of BSCW. in *Computer-Supported Cooperative Work: Special issue on CSCW and the Web, Vol. 6(1997), ©Kluwer Academic Press.*

[4] Microsoft Exchange 2000 Worflow Engine, http://msdn.microsoft.com/msdnmag/issues/0700/exchange/exchange.asp

[5] Anders Kristensen, Formsheet and the XML Forms Language XML Forms reference, in Proceedings of the WWW8 Conference, Toronto, 1999.

[6] Charles K. Ames, Scott C. Burleigh, and Stephen J. Mitchell.WWWorkflow: World Wide Web Workflow, in *Proceedings of the 30th Hawaii International Conference on System Sciences (HICSS-30)*

[7] John S. Davies, IBM MQSeries Workflow: Staff Assignment Techniques, White paper. http://wvvw-4.ibm.com/software/ts/mqseries/workflow/

[8] Lotus Notes, http://www.lotus.com/

[9] Oracle, SQLFlow Workflow Management System, http://170systems.com/SQ/flow.htm

[10] GFI - Email Flow - www.emailflow.com

[11] GFI, Workflow Technology - an introduction (www.workfl.owsoftware.com/workflowwp.pdf).

Two Case Studies for **KAKTUS**

Mark Burnett

DSTO C3 Research Centre
Fernhill Park
Department of Defence
Canberra ACT 2616
AUSTRALIA

Mark.Burnett@dsto.defence.gov.au

Abstract

KAKTUS is a step toward what Tim Berners-Lee, the creator of the World Wide Web, calls a "semantic Web" where people, software agents, search engines and other programs can gain access to meaning—rather than just content—on a Web site. A semantic Web potentially also lets programs utilize all the data on Web pages, allowing them to gain knowledge from one site and apply it to logical mappings on other sites.

Keywords Semantic Web, Entreprise model. Textual recognition.

1 A handle on context

In a world where information environments are becoming larger, more complex and more dynamic, KAKTUS uniquely provides a contextual view of information. KAKTUS recognises that the problem of identifying meaning within text is dependent on the context within which the text is read. It effectively allows different people to have different models of the same data. And by representing the model in an open and transformable way, KAKTUS allows those views, or islands of knowledge, to be shared and exploited within the enterprise.

2 Building a semantic web

KAKTUS builds a semantic web by automatically populating a model of an enterprise, scenario or organisation with data taken from documents. Pattern matching and basic natural language understanding techniques are used for textual object recognition and identification, and provide a low-cost furnishing of the information environment for KAKTUS. These techniques also allow for relatively easy creation, adaption and maintenance of the webs by a trained user or external consultant.

3 Business objects and processes

KAKTUS provides a framework for users to view information through the lens of business objects such as people, places and things and business processes that relate those objects within an enterprise scenario. This is effectively a gearing process that models information at a level that is meaningful to a user, above the unit level of documents or chunks of text.

4 Capabilities

A number of different capabilities or applications can be surfaced from the semantic web. There are applications for information retrieval, information visualisation, knowledge management and fact extraction. The following is an indicative list of possibilities that have already been prototyped.

- Advanced search capabilities that surfaces information based on semantic relatedness to a query term;
- A key list of terms related to a user query based on physical (within-document) and semantic proximity. This can include association of people (as objects within the semantic model) with information;
- Summary of information with respect to a particular topic across a web site or document collection that gives coverage of both the semantic and document dimensions;
- Dynamically hyperlinked web sites, with the hyperlinks relevant to the context and content of the user's information need;
- Extraction of facts - taken here to mean data with a fixed semantic correlation - or partial facts from within a collection of texts.

5 Features

KAKTUS builds a framework for managing information and knowledge with the following attractive end-user features:

- Provides value from day 1 - it doesn't require a "big bang" population of a huge model;
- Scalable - the underlying techniques scale in all directions;
- Flexible - a semantic web can service an entire enterprise over an extended period, or a single analyst for one day;
- Complimentary to existing tools such as search engines

6 Demonstration

Two cases studies for KAKTUS will be demonstrated, illustrating how the approach can be used in diverse scenarios. The first involves planning documents for DSTO for the year 99-00. In this case there's a complex underlying model of the enterprise involving DSTO, Defence, commercial organisations and
their inter-relation.

The second involves PROMED emails. PROMED, the Program to Monitor Emerging Diseases, is the premier discussion group for specialists in emerging infectious diseases. It was proposed by the Federation of American Scientists (FAS) to create a global system of early detection and timely response to disease outbreaks. A relatively simple semantic model underlies the reporting of disease and disease outbreaks for PROMED.

For both case studies we demonstrate: (i) association of key related information following a user query against the document set, based on semantic and physical (within document) proximity; (ii) a collection wide summary of information related to a user-defined topic, that takes account of both key topic coverage and document coverage; (iii) navigation of the information based on dynamically hyperlinked HTML, with links generated in response to the KAKTUS-derived set of related
topics.

For PROMED we also show how an automated approach to fact or partial-fact identification within the emails.

Demonstration of Collaborative Internet-Based Document Development Using Microsoft Tools

Frank Eilert[1], Charles Herring[2], Kate Horsey[3] and Michael Rees[4]

CRC for Enterprise Distributed Systems Technology[1, 3]
Department of Computer Science and Electrical Engineering[2]
The University of Queensland
Brisbane, QLD 4072 Australia

School of Information Technology[4]
Bond University
Gold Coast, QLD 4229,Australia
{eilert, herring, horsey, rees}@dstc.edu.au

Abstract

The demonstration will show how the Microsoft Office family of products and related collaboration tools are used to support a distributed team in developing a document. We will demonstrate generating a workgroup website using a Wizard in FrontPage. The site will be configured specifically to manage a standards type document through a lifecycle. The demonstration will utilize an IIS Server with the Office Server Extensions (OSE) to enable collaboration. Two laptops will be used to simulate the various users. Other collaboration tools will also be involved such as MSN Messenger and NetMeeting.

1 Introduction

The demonstration is designed to show how a collaborative website can be quickly constructed and utilised for the purpose of consultative specification development. Firstly the collaborative site is generated using a FrontPage 'Wizard'. The site can then be modified to suite a particular application. Documents are created and posted on the inter/intranet for comment by the users. The users make comments (discussions) using an Internet Explorer 5 discussion enabled browser. The document is modified based on the discussions and posted to the Web as a final copy.

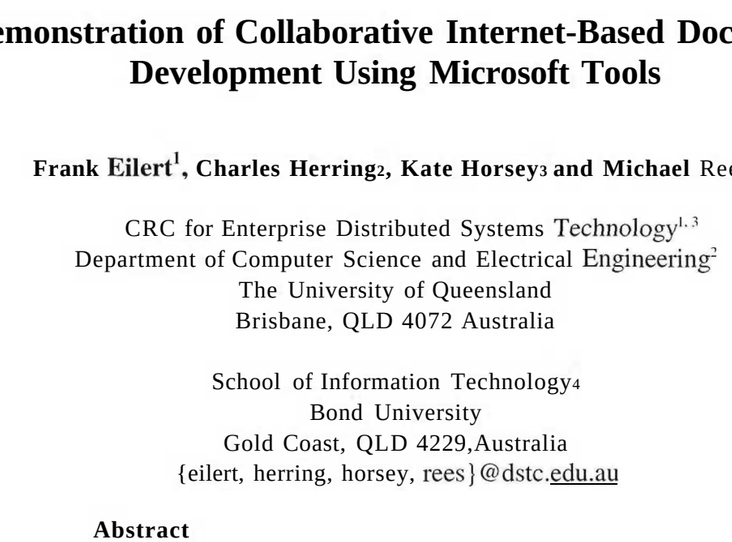


Figure 1 Creating the 60 Minute Web Site

Submitted to 5th Australasian Document Computing Symposium,
Sunshine Coast, Australia,
December 1, 2000.

2. The Demonstration

2.1 Creation of a Workgroup Website

The website is created on a Microsoft Internet Information Server Version 5 **(IIS)** with Office Server Extensions (OSE). The method to create the website is a FrontPage 'Wizard', available from Microsoft called the '60 Minute Intranet Kit". The first step in generating the website from FrontPage, selecting the 60 Minute Intranet Kit Wizard is shown in Figure 1

2.2 Collaborative Document Development

Once the website is operational users may deposit documents and use the various collaborative tools to refine them. These tools include the OSE to the website. Word, Net Meeting and Messenger. In this demonstration the document is created in Word and posted to the working website as a .doc file. and then modified by two of the authors. The demonstration shows how discussions are used asynchronously. Also Messenger and Net Meeting were also used to exchange ideas synchronously.

2.2 Release for Comment

Once the document is developed to a first draft level it is posted to the website as a HTML file. The document is saved to a 'Web Folder' that maps to the directories of the collaborative web site.
People can make comments about and within the document as demonstrated at Figure 2 using IE5. (Other browser can be utilized such as the Netscape Navigator and older Versions of Internet Explorer; however, the functionality to insert discussions in the documents is lost.)

2.4 Closure Modification and Final Posting

After a suitable comment period, a next version of the document can be produced based on the comments made by the users. Replies can be sent to users regarding their comments. The new document posted to the web and another round of comments can be opened if desired.

3 Conclusion

This demonstration scenario is based on a standards document; however the method is applicable to any document management problem that involves consultation with large numbers of users.

References

[1] Microsoft Office 2000. http://www.microsoft.com/office

[2] Microsoft Office Server Extensions. http://www.microsoft.com/office/deployment/default.htm

[3] Microsoft. 60 Minute Intranet Kit. http://www.microsoft.com/office/WebTour.htm

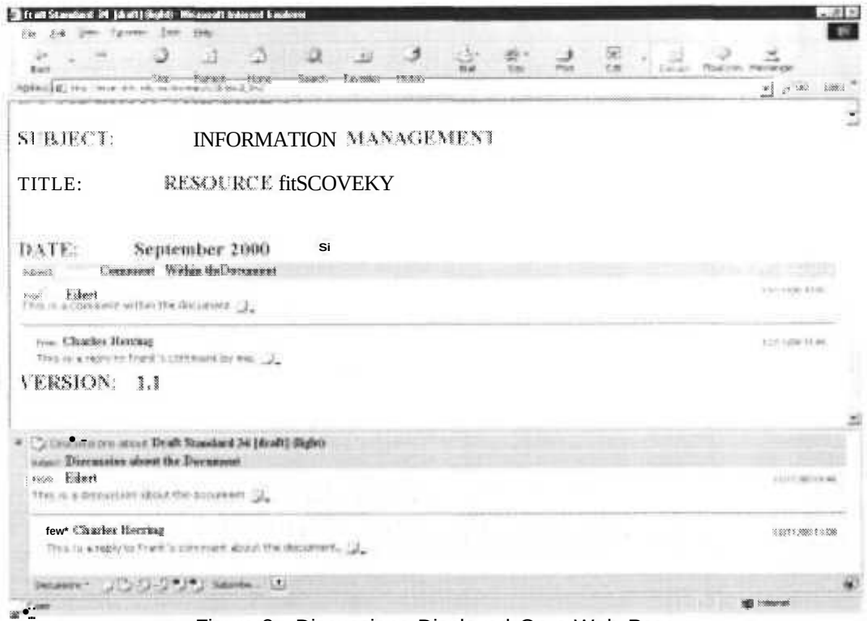


Figure 2 - Discussions Displayed On a Web Page

Machine Mapping of Document Collections: the Leximancer System.

Andrew E. Smith
School of Computer Science and Electrical Engineering
The University of Queensland
Queensland, Australia. 4072.
andrew.smith@member.sage-au.org.au

Abstract

A system is presented for rapidly mapping a text document collection using a set of concept dimensions. This offers an easy way to index and re-index large sets of documents using a flexible, faceted classification system of concepts, rather than just keywords.

Thesaurus-like concepts of interest are specified in advance, along with a few seed words to describe each. These concepts can be selected automatically if desired. A novel machine learning algorithm finds an optimal set. of weighted terms from the document set for each concept. The resulting thesaurus network is used to classify and index the. collection, down to a possible sentence resolution. Weights of concepts and relationships between concepts are measured. and mapped on a concept cluster map. The. concept space can be. explored from an overview, down to the. document level.

Keywords Document Management, Information Retrieval, Text Mining.

1 Introduction

This paper presents a system for generating a conceptual index with minimal supervision. In contrast to normal keyword indexing methods, this system uses abstracted thesaurus concepts, and can offer a more precise and flexible alternative to whole-document classification and filing. In other words, this approach lies between keyword indexing and document classification. with the advantages of both. The method is efficient and requires minimal supervision.

2 Strategy

This project had as its goal the development of a practical Text Mapping and Exploration system. The strategy chosen to achieve this was based on a Concept Space approach (Chen et al [1]), in which words are mapped to a much smaller set of concepts:

1. Text preparation: Standard techniques are employed, including name and term preservation, tokenisation. and the application of a stoplist (eg. Miller et al [2]). No attempt at phrase binding is made. Stemming is not performed since automatic thesaurus generation makes this redundant, and undesirable.

2. Creation of Thesaurus Concepts: These can be devised in collaboration with a domain expert to suit the current requirements of the users. or they can be chosen automatically using a novel algorithm for finding significant seed words to reflect the themes present in the data.

3. Learning the thesaurus: Use a machine learning algorithm to find the optimal thesaurus words from the text data.

4. Classification: Classify the text using this thesaurus. to a possible sentence resolution.

5. Creation of a faceted two-level classification system: Designate primary concepts as entities and secondary concepts as properties of entities: index the tagged text using the entities and properties.

6. Mapping: Cluster the entities according to weight and relationship, to create a Concept Cluster Map.

7. User interface: Generate a simple hypertext browser for exploring the classification system in depth.

3 Machine Thesaurus Construction

The key to achieving concept mapping of text lies in the generation of thesaurus-like mappings from groups of words to concepts. The goal here is to accurately predict the likelihood that a fragment of text represents a subset of the concepts. This is often referred to as tagging.

Proceedings of the Fifth Australasian Document Computing Symposium, Sunshine Coast, Australia, December 1, 2000.

3.1 Techniques for Word Cluster Formation

Traditional document clustering relies on document nearness metrics that use Single Term Indexing (Salton [4] sect 9.3). This treats a document descriptor as a 'bag of words' and ignores local structure. The next logical step is to perform term clustering based on the co-occurrence statistics of words in the collection: to cluster and retrieve documents (Salton [4] sect 9.4), and to construct a thesaurus (Salton [4] sect 9.6).

These Information Retrieval methods focus on document similarity, whereas this project is concerned with conceptual dimensions. This is a key discrimination, for the reason that most term clustering algorithms *induce* pattern features from the text which do not necessarily have clear conceptual meanings. These are useful for matching documents but do not. lead to a conceptual abstraction that can be tailored easily to the interests of the user. A principal goal of the work presented here was to allow the user or the system to tightly define each concept prior to learning and indexing the text body.

3.2 Techniques for Word Sense Disambiguation

As a result of the considerations mentioned in the previous section. the field of word sense disambiguation was examined for suitable methods. This discipline. part of Computational Linguistics. generally seeks to tag words with conceptual category indicators, such as thesaurus classes.

Many researchers have used static thesauri and lexicons to perform sense tagging. One of the most popular thesauri for this purpose is Wordnet (Miller et. al [2]). These methods suffer from the problem that the document vocabulary varies from time to time and from domain to domain. It is expensive to manually update or construct a thesaurus.

A second strand of research uses Bayesian Classifiers (Yarowsky [9]). or more generally. Linear Classifiers (Roth [3]). A Bayesian Classifier is a network of weighted mappings of observed features to likely scenarios. Yarowsky [9] presented a self-ordering algorithm for learning a network to disambiguate the different senses of specific words, such as 'plant'.

We developed a generalisation of Yarowsky's method using the following assertions:

- Words in text fragments constrain the choice of nearby words through the medium of a concept. This idea is borrowed directly from Corpus Linguistics (Stubbs [7]).
- As a result, we can generalise Yarowsky's method from senses of specific words to concepts in general.
- To achieve this, seed a concept with a small set of indicative words, or word combinations, taken from the training data. These words shoud be chosen wisely; the majority should only have meanings contained within the concept class, in the texts under inspection. In practice, a seed set of one word will work fine.

The details of the method are published in Smith [5].

4 Classification

Once the thesaurus network has been learnt, the classification procedure is simple:

1. Process the text sequentially in blocks of n sentences.
2. Look up all the words from the text block in the thesaurus network and add their weightings in each concept.
3. Threshold the results to select the relevant concepts with likelihood weightings.

This is very similar to coding. or sense tagging, as performed in Content Analysis (Weber [8]).

5 User Interface

A prototype system for indexing. mapping, and exploring these classified sentences was developed. The user interface was written in HTML. JavaScript, Java. and CSS. The 'Leximancer' system was designed for browsing the concepts and names present in a body of text. This system uses a two level classification system, where each level is faceted. At the top level are the concepts or names of primary interest. These are called entities. In Information Science terms they are absolute semantic dimensions. Relationship strengths are found in a pair-wise manner between each entity and all the others, which allows the number of entities to be reasonably large while avoiding serious combinatorial explosion.

The second level of the classification system consists of concepts or names which are called properties. These are also absolute dimensions, in that a property can apply to any entity. The properties associated with each entity are treated as facets. Only N-tuple combinations of properties which appear *in the context of each entity* are recorded, again to control combinatorial behaviour. It is emphasised that any concept can be designated as an entity, or a property, or both, depending on the information need of the user.

One of the principal aims is to quantify relationships between concepts. These relationships are then used to allow analysts to explore the text; that is, to approach a document collection like an unfamiliar continent. An overall ma]) of the concept space is created by clustering entities according to the strengths of their relationships. The algorithm used to produce the cluster map allows the concept points to order themselves on the surface of a sphere according to the strengths of the relationships between each pair (an example of the many body problem). The strength of each concept is indicated by its brightness. A screen shot is shown in Figure 1.

By clicking on the entity of interest, or by selecting from the pull down list, the user opens a hypertext browsing panel for that entity in the right half of the window. The entity panel lists the related entities, and the strengths of these relationships. This is a common approach in traditional data mining.

In addition, the associated properties of the concept can be viewed in this panel. In each of these panels which lists the intersections of two entities, or the intersections between an entity and multiple properties, the textual evidence can be browsed by clicking on the button image. Also, the vocabulary used in the text in association with this entity. and in association with each property of the entity, can be viewed.

6 Case Studies

A variety of real-world data sets have been analysed. These include sets of newspaper articles. a 50 Mb sample of Usenet news postings. a 100 Mb collection of job tracking list text data, the novel •Pride and Prejudice'. the King James Bible, and 50 Mb of Federal Court judgements. At the time of writing, most of these case studies are viewable on the web [6]. These experiences have shown that this procedure is successful for learning and classifying from the same body of text. Expert analysts report high recall and precision from their case studies. with simultaneous precision and recall levels of over 90% being recorded. For the larger-data sets, learning was successfully performed on a sample of the data prior to classifying the whole. So far, no suitable benchmarking training data has been found. The Reuters-21578 collection is unsatisfactory because the tags are not embedded near their contextual words.

The algorithms perform efficiently, so mapping and remapping is not difficult. The most demanding phases are learning, indexing, and clustering.

The temporal asymptotic behaviour of the learning algorithm goes as the product of the length of the text body and the number of concepts. Memory usage is roughly proportional to the product of the size of the vocabulary and the number of concepts. The number of iterations for convergence of this algorithm does not depend on the number of words or concepts.

The indexing algorithm is memory limited. No simple proportionality can be assigned to this process. as memory usage depends on the number and nature of the concepts. the division of concepts into entities and properties, and also the textual indexing resolution and threshold chosen. The resolution and threshold can usually be adjusted to suit the available memory.

The temporal asymptotic behaviour of the cluster mapping algorithm is proportional to the square of the number of entity concepts. Memory usage is negligible.

For example, learning 40 concepts from 50 Mb of text. with a vocabulary of 18,000 words and 90,000 names. takes one hour without sampling. This data set has a disproportionately large vocabulary of names. which increases the memory footprint dramatically:

Hardware: Pentium III 500 Mhz single processor with 192 Mb RAM.

Operating System: RedHat Linux 6.1

Memory Footprint: 140 Ml)

Dedicated Runtime: 65 minutes

Cluster mapping 100 concepts takes 30 minutes. This is probably the largest number of points that should bo presented on one chart for cotnprehensibility:

Hardware: Pentium III 500 Mhz single processor with 192 Mb RAM.

Operating System: RedHat Linux 6.1

Memory **Footprint:** less than 10 Mb

Dedicated Runtime: 30 minutes

The final learning state is saved and this can be used next time a similar document set is analysed using the same concepts. This makes convergence much more rapid. Also. additional concepts can be learned at a later time and added to the thesaurus network.

7 Conclusion

It is proposed that, in some applications, such a process of conceptual mapping of document sets could be a better solution than filing or indexing. Conceptual mapping can offer less uncertainty to the user than keyword indexing, thus achieving better recall and precision. In contrast to whole document classification, mapping does not have to assign the same concepts to an entire document. More importantly, the technique presented here is relatively fast and largely unsupervised.

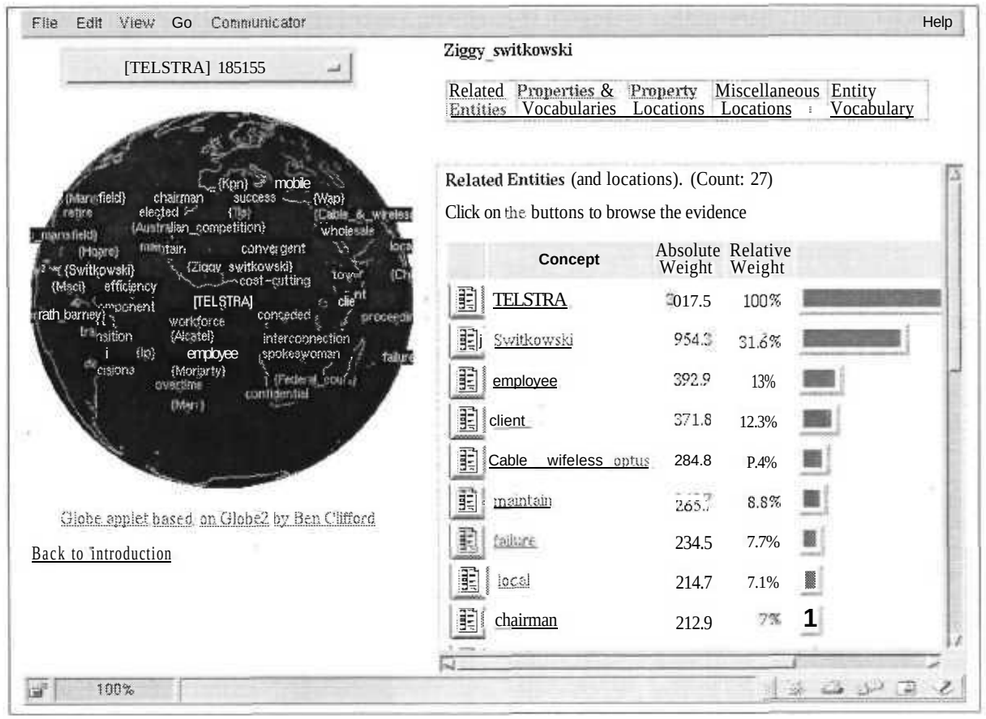


Figure 1: A screen shot of Leximancer.

References

[1] H. Chen. B.R. Schatz. T.D. Ng. J.P. Martinez, A.J. Kirchhoff and C. Lin. A parallel computing approach to creating engineering concept spaces for semantic retrieval: The illinois digital library initiative project. *IEEE Transactions on Pattern Analysis and Machine Intelligence. Special Section on ••Digital Libraries: Representation and Retrieval",* Volume 18, Number 8. pages 771 782. August 199G. http://ai.bpa.arizona.edu/go/report/technicalReport.html.

[2] George A. Miller, Richard Beckwith. Christiane Fellbaum. Derek Gross and Katharine .1. Miller. Introduction to wordnet: an on-line lexical database. *International Journal of Lexicography.* Volume 3. Number 4, pages 235-244. 1990. http://www.cogsci.princeton.edu/~wn/.

[3] Dan Roth. Learning to resolve natural language ambiguities: A unified approach. In *Proceedings of the Fifteenth National. Conference on Artificial Intelligence (AAAI-98),* pages 806-813, Madison, Wisconsin, 1998. http://12r.cs.uiuc.edu/-danr/publications.html.

[4] Gerard Salton. *Automatic Text Processing.* Addison-Wesley, 1989.

[5] Andrew E. Smith. Machine learning of well-defined thesaurus concepts. In *Proceedings of the International Workshop on Text and Web Mining (PRICA1 2000).* pages 72-79. Melbourne. Australia. August 2000.

[G] Andrew E. Smith. Text mining. automatic classification and indexing. http://www.csee.uq.edu.au/~aes/, July 2000.

[7] Michael Stubbs. *Text and corpus analysis : computer-assisted studies of language and culture.* Blackwell Publishers, 199G.

[8] Robert Weber. *Basic Content Analysis.* Sage Publications. 1990.

[9] D. Yarowsky. Unsupervised word-sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL-95).* pages 189–196. Cambridge. MA. 1995. http://www.cs.jhu.edu/~yarowsky/pubs.html.

Presentation of Tabular Information on WAP Enabled Devices.

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

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

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

Abstract.

The limited size of display available on the majority of WAP devices means that its not possible to effectively display information stored in tabular structures. In this paper we present an approach that exploits the hierarchical nature of tabular information to enable table content to be decomposed over a number of WML cards, providing access to content in an effective and intuitive manner.

Keywords: Document standards, WAP, WML.

1. Introduction.

Mobile computing is a rapidly expanding area in which the recent release of WAP (Wireless Application Protocol) compatible mobile phones has gained significant attention. WAP phones (and other enabled devices) are intended to provide mobile access to information sources in a similar manner to the Internet. One problem that we envisage with such devices is that the limited size of display available may restrict the types of information that can be effectively displayed. A prime example of this is information stored in tables.

The nature of tabular layout means that it is not suitable for rendering on a small display. In order to overcome the problems imposed by a small display, we propose an approach that exploits the hierarchical nature of table content to enable effective and intuitive display of tabular information on WAP enabled devices.

2. WAP and WML.

In the same way that HTML (Hyper Text Markup Language) is used to serve content to Web browsers, content for WAP devices is defined in WML (Wireless Markup Language) [1]. Unlike HTML, WML is designed to work with compact wireless devices that by their nature have small displays, limited computational resources and low bandwidth high latency networks.

All WML information is organized into a collection of cards and decks (as illustrated in Figure 1). In WML the deck is the smallest unit that a server can send to a WAP device. Each deck contains one or more cards and it is possible to navigate between cards and decks via choice options, links etc, in much the same way as with HTML documents. Sending a deck of cards, as opposed to a single card, helps overcome some of the associated latency problems as a user can browse and navigate through cards within a deck without the need to make requests to the WML server.

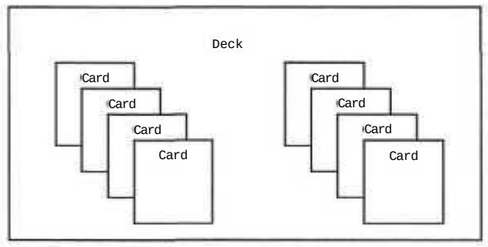


Figure 1: A WML Deck and Cards.

3. Displaying Tabular Information on WAP Devices.

The restricted size of displays on WAP devices means that it is often difficult to display information in an appropriate manner. Content created for WAP devices must therefore be as compact as possible, often limiting the type of information that can be supported. This is particularly true for tabular information. Whilst WML provides support for tables, only small tables (maximum 5x5 cells) with a simple grid structure are supported [2].

This is unfortunate as a great deal of information that may be useful for WAP users is stored in a tabular fashion (whether it is in plain text or HTML tables, or stored in a DBMS or spreadsheet). Fortunately, due to the hierarchical nature of the content of many tables, it is possible to display this information in a way that is not only clear and logical but also conforms to limitations of screen size and memory imposed by WAP devices.

3.1 The Hierarchical Nature of Tables.

Many tables contain data that is divided into groups or classifications. These groupings usually manifest themselves as repeated or spanning entries in the left most columns of a table. The nature of these groupings results in a table that can offer a simple and intuitive way for a reader to locate specific data, by selecting only the relevant subset of data at each phase of their search, finally focusing on the information of interest. To illustrate this, consider Figure 2, which shows a table containing information about Hifi components. If a reader wishes to locate information about a specific component, (in this case a specific Sony CD player) they would do so by locating the information relating to the appropriate make, then the type of component and finally the model of interest. The information in focus at each stage of this process is indicated.

Where information in tables contains groupings in this way, the content can be modelled as a tree as shown in Figure 3. A hierarchy is formed based on the groupings and each leaf node contains an item of data relating to a specific entry

Modelling the content of a table in this way is highly suited to WML. By creating the equivalent hierarchy in a series of WML cards, the content of each card can be kept small enough to be comfortably viewed, whilst providing access in a logical manner. The hierarchy is modelled as follows:

Make	Component	Model	Price	Test Date
Sony	Amp	TA FE2 3 0	100	Dec 99
Sony	Amp	TA FE330R	130	June 00
Sony	CD	CDP XE22 0	100	April 00
Sony	CD	CDP XB 93 0E	300	Sept 99
Sony	Tuner	ST SE300	120	April 99
Marantz	Amp	PM 4000	150	June 00
Marantz	CD	CD 4000	100	April 00
Marantz	CD	CD 6000 OSE	300	Sept 99
Marantz	Tuner	ST 77	600	May 98

Figure 2: How hierarchy within a table guides access.

- A separate card (a choice card) is generated for each set of choices. Each of the choices at each stage is composed of a link (just like a hyperlink in HTML) which connects to the next card in the hierarchy (whether it be another choice card or a data card containing information relating to an entity).
- A data card is generated for each entity, containing appropriate data from the leaf nodes.

Figure 4 shows the WML cards that would be displayed for each set of choices in our example. The arrows indicate the links that exist between the cards.

3.2 Effective Access to Table Content.

Because of the fragmented nature of the presentation of table content, it is essential that the user is able to guide the focus of the search. For example, when searching for Hifi components in the table in Figure 2, if the reader was specifically looking for details of CD players, it would be useful to have component types at the root card. With any other information al the root, the user would need to perform an exhaustive search, traversing a large number of links (often unnecessarily) in order to locate the appropriate details. To enable effective access, we allow information in any of the columns that display groupings to be chosen as the start point for the search (i.e. the root card).

3.3 Generating WML.

In this section, we shall discuss how the WML required to represent a table is produced. Two basic types of card are generated:

- Choice cards - generated at each stage where a choice should be made. There are two types of choice card, one for the root and another for all other choices, which enables navigation to the previous card.

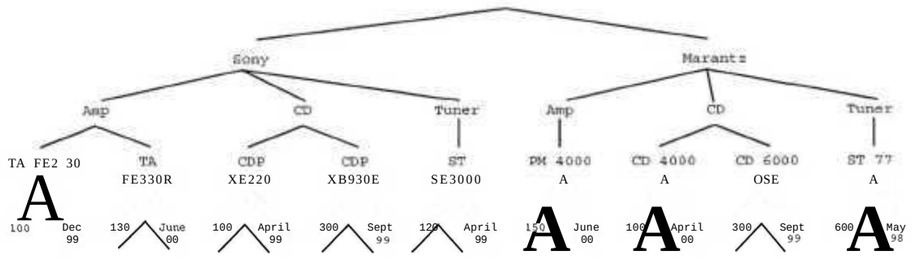


Figure 3: Tree representation of table content.

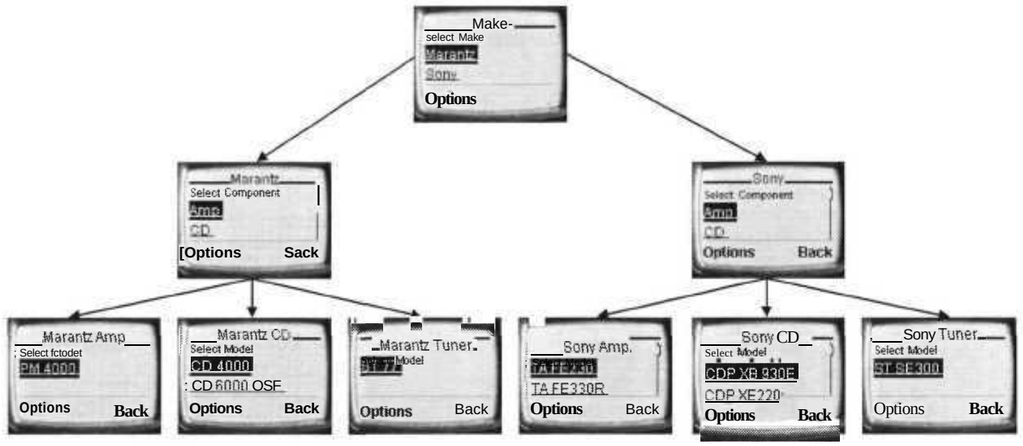


Figure 4: Hierarchy of WML 'Choice' Cards.

- Data cards - generated for each entity, containing information that appears in the appropriate row of the table.

We shall illustrate the generation of the required WML by examining code segments for each type of card. In each case, bold type indicates any information imported from the source table.

3.3.1 The Root Card.

Figure 5 shows the WML generated for the root card. The first line of a card definition contains a `card id`, which is a unique name for the card, and a title to be displayed. For the root card, these are simply set as the label for the first column of the source table (i.e. Make in this case).

On line 4 the user is prompted to make a choice and again the label of the first column is used. This prompt is followed by each of the choices available at this level.

```
1       <card id = "card_Make" title="Make">
2
?               <p>
4                       <small>Select Make</small>
S                       <br/>
6                       <anchor title="Marantz">Marantz
                                <go href="#card_Marantz"/>
8                       </anchor>
9                       <br/>
10                      <anchor title="Sony">Sony
11                              <go href="#card_Sony"/>
12                      </anchor>
13              </p>
14      </card>
```

Figure 5: WML source and resulting display for the Root Card.

For each choice, a link (href) is generated where the title consists of the choice field (line 6) and the href consists of this choice field concatenated after '`#card_`' (line 7).

3.3.2 Choice Cards.

The remaining choice cards are generated in a similar manner to the root card. Figure 6 shows the WML generated for such a card. The main difference to note is the extra code that allows the user to navigate backwards through the cards visited (lines 3-5).

In addition, the `card id` and `title` are more complex. In choice cards beneath the root, these consist of a concatenation of all nodes visited to reach this stage. For `card id` these are simply joined (as spaces are not valid in `card id`) and concatenated after '`card_`', whilst for `title`, spaces are indented between each token (to aid readability).

As with the root card, the selection prompt (line 8)


```
1       <card id = "card_MarantzCD" title="Marantz CD">
2
3       <do type="prev" name="prev" label="Back">
4               <prev/>
5       </do>
e
7               <p>
8                       <small>Select Model</small>
9                       <br/>
10                      <anchor title="CD 4000">CD 4000
11                              <go href="#card_MarantzCDCD4000"/>
12                      </anchor>
13                      <br/>
14                      <anchor title="CD 6000 OSE">CD 6000 OSE
15                              <go href="#card_MarantzCDCD6000OSE"/>
16                      </anchor>
17              </p>
18      </card>
```

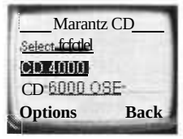


Figure 6: WML source and resulting display for remaining Choice cards.

utilises the appropriate label for the data used to generate the choices.

Again, links (hrefs) are generated for each choice. In a similar way to the `card id`, the reference is generated using a concatenation all nodes visited to reach this stage. This is joined to '`#card_`' and finally to each of the available choices (see line 11).

3.3.3 Data Cards.

A data card is created for each row of the source table. The `card id` and `title` are created in the same way as for choice cards. For this illustration, the data cards are defined to contain all information from the source table. Thus in Figure 7, the content (defined in lines 8-12) consists of each label and the associated attribute value

4. Conclusion.

By exploiting the hierarchical nature of information within tables, it is possible to present the information they contain in a compact and intuitive format suitable for viewing on the restricted displays of WAP devices. Based on this concept, we have developed a software tool that can transform tables from plain text and HTML documents, spreadsheets and DBMSs into appropriate WML decks.

5. References.

[I] Nokia WML Reference, Version 1.1, Available from http://www.forum.nokia.com

[2] The Independent WAP/WML FAQ, Available from http://wap.colorline.no/wap-faq/

[3] L.E.Hodge, W.A. Gray and N.J. Fiddian. Effective Reuse of Textual Documents Containing Tabular Information. *Proceedings of the Fourth Australasian Document Computing Symposium (ADCS 99)*, Coffs Harbour. NSW. Australia. December 1999.

[4] L.E.Hodge, W.A. Gray and N.J. Fiddian. A Toolkit to Facilitate the Integration of Tabular Information in Textual Documents with Database Applications. *Proceedings of The Fourth Multiconference on Systemics, Cybernetics and Informatics (SCI 2000)*, Orlando, Florida, USA, July 2000.

```
1       <card id = "card_MarantzCDCD6000OSE" title="Marantz CD CD 6000 OSE">
2
3       <do type="prev" name="prev" label="Back">
4               <prev/>
5       </do>
ft
7               <P>
8                       <b>Make:</b>Marantz<br/>
9                       <b>Component:</b>CD<br/>
10                      <b>Model:</b>CD 6000 OSE<br/>
11                      <b>Price:</b>300<br/>
12                      <b>Test Date:</b>Sept 99<br/>
13              </p>
14      </card>
```

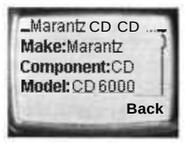


Figure 7: WML source and resulting display for a Data card.

Consultative Standards Development using Microsoft Office 2000 and Office Server Extensions

Frank Eilert[1], Charles Herring[2], Kate Horsey[3] and Michael Rees[4]

CRC for Enterprise Distributed Systems Technology[1,3]
Department of Computer Science and Electrical Engineering[2]
The University of Queensland
Brisbane, QLD 4072 Australia

School of Information Technology[4]
Bond University
Gold Coast, QLD 4229,Australia
{eilert, herring, horsey, rees}@dstc.edu.au

Abstract

The Communications and Information Policy & Planning section (CIPP) of the Department of Communication, Information, Local Government, Planning and Sport (DCTLGPS), Queensland State Government, required a solution to electronically manage the processes and documents associated with the preparation. consultation, release and maintenance of Information Standards. The solution needed to provide a best practice model for other government agencies, to comply with the Government Information Architecture, and utilise existing infrastructure. The team of authors of this paper, from DSTC Pty Ltd, proposed a solution utilizing several standard Microsoft tools. This solution has been trialled and the experience and project outcomes are described in this paper.

Keywords Document management. group discussions, Microsoft Office 2000, office server extensions, processes for comment.

1 Introduction

The Communications and Information Policy & Planning (CIPP) section of the Department of Communication, Information, Local Government, Planning and Sport (DCILGPS), Queensland State Government (QG) required a solution that would enable them to electronically manage the processes and documents associated with the preparation, consultation, release and maintenance of Information Standards (IS). The solution needed to provide a best practice model for other government agencies and to comply with the Government Information Architecture while utilising existing infrastructure where possible.

The CIPP staff wished to work collaboratively with their client agencies over the Internet to develop standards documents. The original process was manual, although the documents were posted to a web site. The key driver was to have 'one pass' consultation on the draft standard before it is finalised. To accomplish this CIPP required an environment where all the agencies in the review process could not only review the document but also review proposed amendments from other agencies. CIPP also sought a tool that allowed effective online management of the overall document development process.

DSTC proposed a solution utilizing the Microsoft Office 2000 suite [1] and the Office Server Extensions for Internet Information Server (IIS) tools [2]. Also, the "60 Minute Intranet Kit" [3] was used by CIPP to manage its internal organization of the standards documents. The initial implementation was designed be a "straight out of the box" implementation of the software. Some re-engineering of processes was also required to ensure the tools would support the business process in the most effective manner.

2 Standards Development in DCILGPS

2.1 Requirement

Information Standards are developed by CIPP for Queensland Government agencies. The standards attempt to give consistency of IT usage across thirty-eight agencies within Queensland Government. Effective uptake and implementation of the standards relies on adequate consultation with all agencies. The current system requires comments returned via e-mail or mailed paper copies. This method is time consuming for CIPP staff, as they are required to collate the comments and resolve comments from different agencies. CIPP required a system where the agencies could efficiently comment and discuss the

Proceedings of the 5th Australasian Document Computing Symposium,
Sunshine Coast, Australia,
December 1, 2000.

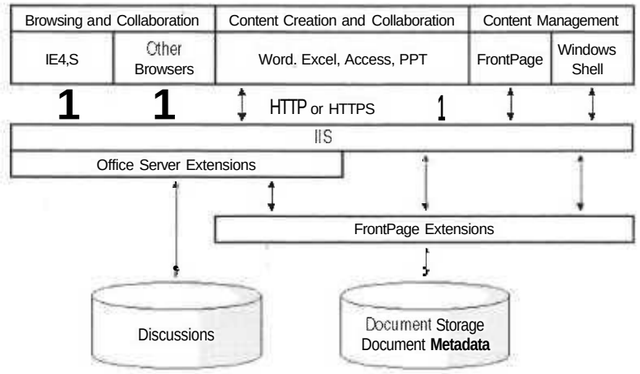


Figure 1 Office Server Extensions Client and Server Architecture

standards on-line, in a transparent context with multiple contributors. The first step was to define a business process that would effectively utilise the affordances of the new tools. The business process was not a paradigm shift but a modification of an effective manual process. The new process is described below.

2.2 Business Process

The business process that Information Standards follow from development through to approval is supported by the collaborative tools. The process consists of the following three distinct stages:

Initiation –Where a need is identified for a new Information Standard a project brief is developed for approval. After approval, CIPP staff prepare a draft of the new Information Standard. This document is posted to the collaborative site and a working group discusses and collaborates on it until a final Working Draft is developed.

Comment - The Working Draft is approved for publication and comment and a new document, the Request for Comment (RFC) is created. The RFC is published to the GIA Web for a period of comment and collaboration with Information Standards Officers from within each Queensland Government agency. Based on these discussions CIPP staff prepare the final Information Standard for approval.

Release - When the Information Standard is approved for release it is published to the CIPP Public Website and agencies are notified that a new Information Standard is in force.

To provide an Internet-based collaborative solution to the above described IS development process, we proposed the use of several Microsoft tools. These are described in the next section.

3 Microsoft Collaboration Support Tools

Our solution to CIPP's consultative standards development requirements uses the following tools: Microsoft Office 2000 [1], Office Server Extensions [2] for Internet Information Service (Microsoft's web server software), and the "60 Minute Intranet Kit" [3] for internal collaboration and organization of IS. These products allow documents to be posted to a web site (using Web Folders), permit users to make comments (discussions) about the document, and to be notified of changes to documents (email notifications). These tools and features provide the basis for our solution to CIPP's Internet-based consultative standards development requirements.

3.1 Consultative Development

The Office Server Extensions (OSE) architecture uses both client-side components and server-side services. For the purpose of the DCILGPS project the client components, Office 2000, have been deployed within the CIPP environment. However, agency clients are not necessarily configured with Office 2000. Agency clients access to the documents is via a web browser and this may not necessarily be the IE5 browser that supports discussions. Figure 1 shows the overall architecture of the IIS/OSE components. It can be used to describe our procedure for managing documents in the discussions environment.

The system functions in the following steps:

1. The Internet Information Server (IIS) is installed and configured to allow discussions.
2. A workgroup website is created utilising the "60 Minute Intranet Kit", a FrontPage 2000 wizard that creates the web site. This website is used internally by CIPP to manage the overall document development process. (See Section 3.2)
3. An author creates a document and stores it in the appropriate Web Folder (Content Creation and Collaboration box, Figure 1). The wizard creates a web folder and a user of Office 2000 can access the folder as long

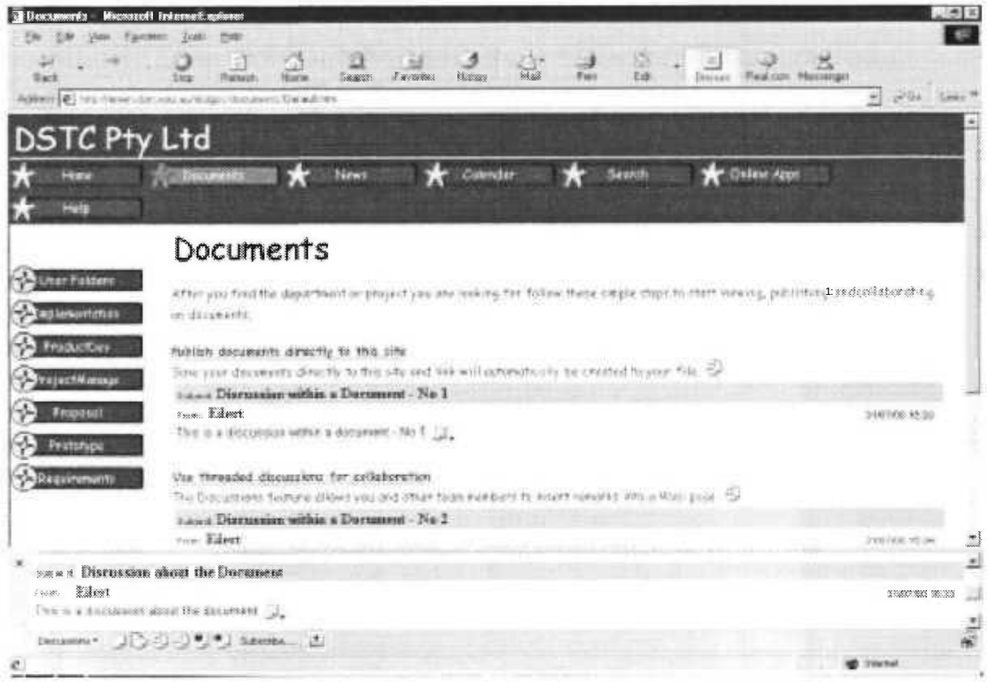


Figure 2 Work Group Web Site Showing Document Discussions

as they have appropriate permissions. The document is stored in the Document Storage area (see Figure 1).

4. The user accesses the page displayed via the web server (Browsing and Collaboration in Figure 1). Any discussions entered are stored in the discussions database, which is separate to the actual document storage. Users may subscribe to be notified of any change to documents or folders. Figure 2 shows how discussions appear in IE. Notice both "in-document" and "about-document" discussion items are shown.

3.2 Internal Development Support

In addition to the support for external agencies to comment on standards under development, CIPP also required support for the overall standards document development process and the "60 Minute Intranet Kit" was used for this. This kit is an advanced wizard that can be added into FrontPage 2000. It permits a user to rapidly (in 60 minutes) develop a team work group web site. The wizard asks a number of basic questions about how the team is to organize its documents, related information and information sources. Based on the answers an integrated web site facility is generated. There are places for all material related to specific aspects of the project and for each user. Figure 2 shows the results. This is a work group web site generated for the project team in order to manage the DCILGPS project itself.

3.3 Client Software Requirements

The Office and browser software versions determine the quality of the user experience. For basic functionality, the client computer requires only a Web browser. Users with Netscape Navigator or Internet Explorer 3.0 can access OSE discussion and subscription features via a frames-based version of the Discussions toolbar. Users with Office 2000 and Internet Explorer 4.01 or higher can access OSE features directly from an Office application or the browser.

4 Conclusions and Future Work

4.1 Outcomes

The first stage of the implementation has enabled the CIPP staff to effectively collaborate on their own documents. Given they are running MS Office 2000 and there were no security issues, this has proven to be simple to implement and successfully met CIPP's needs. CIPP staff could start using the system straight away with very little training. Issues that arose from this phase of the implementation related to saving the discussions within the document for future reference.

The second stage of the implementation enabled access from outside DCILGPS to Information Standards. This task is ongoing as groups of users are trained. Initial feedback from the users is positive and favourable comments were made about the speed at which the process can now take place. The discussion tools also seeded ideas on how processes could be enhanced to further increase efficiency. Work has begun on managing an Information Standard through its lifecycle at the time this article was written. This work will provide more feedback, however, it is expected that this feedback will focus more on the process than the tool.

4.2 Future Work

The initial work has demonstrated the effectiveness of the Microsoft tools for consultative standards development.

Future enhancements may include:

- Extracting discussion text from the SQLServer database. providing enhanced discussion annotation features and merging original document text and discussion text for archival purposes.
- Tracking of issues related to specific information within the document.

Additional Microsoft tools that could be used to enhance the collaboration are:

- MSN Messenger[4] for short real-time discussions
- NetMeeting[5] for real-time discussions of complex issues that cannot be resolved using Messenger
- Digital Dashboard[6] for succinct display of project progress.

These tools can be installed quickly at low or zero cost to the organisation, yet they can provide considerable enhancements to the business processes with minimal training.

Acknowledgements

The work reported in this paper has been funded in part by the Co-operative Research Centre Program through the Department of Industry, Science & Tourism, Australia.

References

[1] Microsoft Office 2000, http://www.microsoft.com/office

[2] Microsoft Office Server Extensions, http://www.microsoft.com/office/deployment/default.htm

[3] Microsoft, 60 Minute Intranet Kit, http://www.microsoft.com/office/WebTour.htm

[4] MSN Messenger, http://ninemsn.com.au//

[5] Microsoft NetMeeting. http://www.microsoft.com/windows/netmeeting/

6] Microsoft Digital Dashboard Kit, http://www.microsoft.com/solutions/km/DigitalDashboard.htm