Ma 64 UDC 81.327:159.95:519.683.5:681.326.34

ACTA POLYTECHNICA SCANDINAVICA

MATHEMATICS AND COMPUTER SCIENCE SERIES No. 64

On Domain Modelling for Technical Documentation Retrieval

PASI TYRVÄINEN

Nokia Research Center
Software Technology
Heikkiläntie 7
B.O. Pox 45, FIN-00211 Helsinki, Finland

Dissertation for the degree of Doctor of Technology to be presented with due permission for public examination and debate at Helsinki University of Technology (Espoo, Finland) on the 15th of March, 1994, at 12 o'clock noon.

HELSINKI 1994

Tyrväinen, P., **On Domain Modelling for Technical Documentation Retrieval,** Acta Polytechnica Scandinavica, Mathematics and Computer Science Series No. 64, Helsinki 1994, 165 pp. Published by the Finnish Academy of Technology. ISBN 951-666-406-7. ISSN 0355-2713. UDC 681.327:159.95:519.683.5:681.326.34

Keywords: Domain modelling, information retrieval, technical documentation, product information management, knowledge representation and transfer, thesaurus construction, hypertext

ABSTRACT

This work addresses the problem of transferring knowledge of large technical systems from the designers to the end users. The proposed solution is to aid retrieval of technical documents by constructing conceptual models to describe the domains of the documentation. The technical focus is on reducing the human effort needed to construct these domain models. First, different text retrieval approaches and the use of domain modelling in them are analysed. For this purpose, a framework of representations used in text retrieval systems is devised. The analysis covers the information retrieval (IR), the natural language processing (NLP), and the hypertext and related document structure approaches. On the basis of this general analysis, the special features of the technical documentation process, text retrieval, and domain modelling in technical domains in particular are discussed. As a result, a list of requirements for ideal knowledge representations aimed at supporting text retrieval and a general principle guiding domain modelling processes in technical documentation domains are proposed. The realisation of the methods is based on the D&T domain models and the DTM modelling system developed in the Esprit II project SIMPR. The D&T models satisfy the requirements presented earlier and emphasise explicit representation of central domain structures to the users. The DTM modelling system supports the creation of conceptual domain models based on available design information databases, utilises structural homogeneity of the information, and enables integration of the models with various retrieval techniques to map the models with texts. This aproach leads to a solution with which it is possible to create complex hypertext-like structures with minimal human intervention by filtering relevant information from existing design data sources. The approach is validated in an industrial pilot.

FOREWORD

This research is mainly based on the work performed in close connection with the SIMPR project during the years 1989 to 1992, although contains some parts from other related work of the author. SIMPR (Structured Information Management: Processing and Retrieval) was the project number 2083 of the second phase of European Strategic Program in Information Technology (ESPRIT II). The work of our project group at Nokia Research Center was partly sponsored by the Technology Development Centre in Finland (TEKES). I wish to express my sincere gratitude to these organisations for providing us with extremely satisfactory working conditions as well as to Professor Markku Syrjänen from Helsinki University of Technology for supervising this dissertation.

The conceptualisation and implementation of the ideas described in this thesis would not have been possible without the enthusiastic spirit within our project team. I owe the most grateful thanks to Mr Petteri Saarinen and Mr Kimmo Hätönen for the constant brain-storming and debates and to Dr Pertti Lounamaa for providing the exceptional opportunity to work in a long term research project like this.

The members of the 9 SIMPR project groups from 6 European countries I wish to thank both for the fluent and educating co-operation in the joint venture and for their warm hospitality during the workshops. Special thanks go to Mr Godfrey Smart, the primus motor of the project well as to Professor Fred Karlsson and Dr Atro Voutilainen at the University of Helsinki for the fruitful co-operation. The staff at Customer Documentation at Nokia Telecommunications Switching Systems I wish to thank for their open attitude in adapting the view of our work giving us the unusual and extremely satisfying position to see the results of our research to develop into a production system.

I thank also all the people providing me the feed-back needed for crystallising the ideas presented in this work by spending their time on reading and commenting it. Special thanks go to Associate Professor Kalervo Järvelin, to Professor Peter Ingwersen, and to Dr Alain Smeaton for their constructive criticism. Translator Tarja Suorsa I thank for checking the language of this thesis.

During this research I gave up the belief on power of words, that they can adequately describe the concepts in technical systems. I will be forever grateful to my wife Marianne on teaching me that even in much more complicated domains it is still worth while to give them a try.

Helsinki, February 1994

Pasi Tyrväinen

CONTENTS

1 Introduction 7
1.1 Background 7
1.2 Assumptions and Objectives 10
1.3 Scope 11
1.4 Contents of the Thesis 11
2 Text Processing and Retrieval 14
2.1 Central Concepts 14
2.2 Historical Background 16
2.3 Early Retrieval Systems 17
2.4 Thesaurus Construction 19
2.5 Text retrieval Problems 21
2.6 Artificial Intelligence 23
2.7 Natural Language Processing 24
2.8 Document Structures and Hypertext 25
3 A Proposed Framework for Representations 28
3.1 Introduction 28
3.2 Strategies 28
3.3 Levels of Representation 30
3.3.2 Knowledge Representations 31
3.3.3 Text Representations 32
3.3.4 Mappings 33
3.4 A Framework for Representations 34
3.5 Other Dimensions of Text retrieval 35
4 Analysis and Overview of Text retrieval Approaches 38
4.1 Conventional Approaches 38
4.2 Natural Language Processing Approaches 41
4.2.1 NLP-Based Text retrieval 41
4.2.2 Problems with NLTR 45
4.3 Structure-Oriented Approaches 48
4.3.1 Mark-up and Document Structures 48
4.3.2 Hypertext 51
4.3.3 DMG -- Hypertext Generation 53
4.4 Intelligent Databases 55
4.5 Evaluation of Alternative Approaches 56
4.6 Discussion 57
5 Technical Documentation 59
5.1 Purpose of Technical Documentation 59
5.1.1 Documentation -- Corporate Knowledge 59
5.1.2 Text vs. KR as a Medium 60
5.2 Technical Documentation 61
5.2.1 Documentation and Product Life-Span 61
5.2.2 Language and Structure of Documents 62
5.2.3 Design Databases and Documents 63
5.3 Text retrieval Process in Technical Domains 63
5.3.1 Problem-Oriented Text retrieval 63
5.3.2 Cost and Result of the Process 65
5.3.3 Effort/Result Diagrams 67
5.4 Selecting TR Methods 69
5.4.1 Contribution vs. Costs 69
5.4.2 Costs of a Text retrieval System 71
6 Domain Modelling in Technical Domains 74
6.1 Special Features 74
6.1.1 Artifact Warrant Principle 74
6.2 Existing Approaches 78
6.3 Utilisation of Design Information 81
6.4 Knowledge Representations for Domain Modelling 82
6.4.1 Requirements for a KR 82
6.4.2 Shortcomings of Current Approaches 85
7 D&T Models -- A Knowledge Representation Supporting Text retrieval 87
7.1 Approach 87
7.2 Domain and Task Models 88
7.2.1 Definition Scheme 89
7.2.2 Instance World 90
7.2.3 View Types and Type Hierarchy 90
7.2.4 Text retrieval using D&T Models 91

7.2.5 An Example 91
7.2.6 Formal Description of D&T Models 94
7.3 Uses and Users of D&T Models 96
7.3.1 Position in the Framework 96
7.3.2 Purposes 97
7.3.3 User Roles 97
8 DTM -- Semi-Automated Domain Modelling for Technical Domains . 99
8.1 Size of D&T Models 99
8.2 CIOS Text retrieval Architecture 101
8.3 Creation of D&T Models 102
8.3.1 Introduction 102
8.3.2 Definition Scheme vs. Instance World 103
8.3.3 Importing Design Information 103
8.3.4 Operation 104
8.4 Creation of Text References 105
8.4.1 Direction of Creation 105
8.4.2 Dynamic and Fixed Text References 106
8.4.3 Examples of Text references 106
8.4.4 Text Reference Classes 109
8.5 Workflow of D&T Modelling Process 111
8.6 Guidelines for Applying DTM 113
9 Evaluation of Proposed Solution in a Pilot 118
9.1 TeleSIMPR Pilot 118
9.1.1 Document Processing 118
9.1.2 Domain Modelling 120
9.1.3 Text retrieval 122
9.2 Case 2 - IGE Software Library 123
9.2.1 The Domain 123
9.2.2 Modelling IGE Domain 124
9.2.3 Advantages and Draw-Backs 126
9.3 Case 3 - Visual Planner Graphical Tool 128
9.3.1 The Domain 128
9.3.2 Modelling the Domain 128
9.3.3 Advantages and Draw-Backs 129
9.4 Evaluation of the Approach 131
9.4.1 Views to Evaluation 131
9.4.2 Artifact Warrant Principle 131
9.4.3 DTM and KR Requirements 133
9.4.4 DTM and Requirements for Modelling 135
9.4.5 Scalability and Limits of DTM 137
9.4.6 KR-Oriented View to Text retrieval 139
9.4.7 DTM and Other Systems 139
10 Summary 143
11 Conclusions 145
12 Future Research 146
12.1 Extensions to D&T Models 146
12.2 Dynamic Models 147
12.3 E/R Diagrams and Performance Evaluation 148
12.4 Task Sequence Modelling 150
References 152
Appendices 165

1 INTRODUCTION

1.1 Background

Increasing complexity of technical systems has increased the need to transfer large bodies of knowledge to human users operating the systems. In general, technical information is transferred primarily through personal contact, especially by turnovers serving as human carriers of knowledge. As a characteristic example, it has been reported that no evidence can be found of successful transfer of space technology by any other mechanism than turnover, despite the many techniques that have been tried and the substantial amount of money invested in promoting the transfer. This phenomenon is based on the fact that ideas have no existence outside the minds of human beings. Ideas can be represented in verbal form, but such a representation is inherently incomplete and cannot be easily structured to fit new situations. For a truly effective transfer of technical information, we must make use of the human ability to recode and restructure information so that it fits into new contexts and situations. Consequently, the best way to transfer technical information is to move the human carrier. [6, pp. 40-43]

In practice, the knowledge and information needed to manage and operate large technical systems cannot be distributed economically by human beings because of the limited number of knowledgeable human carriers and the costs and time needed to teach new experts. The economically feasible alternatives for duplicating and distributing knowledge in contemporary technical environments include 1) the use of formal knowledge representations embedded in expert systems or knowledge-based systems 2) the use of natural language in the form of technical documentation, and 3) the use of combinations of these two.

The research on cognitive science and knowledge representation has equipped us with formal techniques to represent knowledge. These techniques are often used in form of expert systems and knowledge-based systems that perform or aid the users to perform a specific task with a product, such as configuring a computer system, diagnosing faults, etc. The major disadvantage of this approach is the knowledge acquisition bottle-neck; the effort needed to create an expert system or a knowledge-based system is usually tolerable only for very focused applications performing a specific task.

Text is one of the most common form of storing and communicating knowledge; for example, in books, reports, and manuals. Corporate manuals and documentation contain large proportion of the fundamental knowledge of an organisation. Text can be a very effective way of storing knowledge since it is, in general, easy to prepare and straightforward to understand. However, the richness of natural languages can lead to ambiguities and inconsistencies, and authors are often required to possess considerable language skills to be able to ensure that the text will clearly convey to the reader the

meaning intended. [147] As technical documentation is only an incomplete by-product that is often most useful only when the author is directly available to explain and supplement its content [6, pp. 2-5], the use of knowledge representations (or other similar means) to aid the reader are especially important in technical domains.

The use of computerised document management systems and electronic archives is claimed to provide better storage, management, transmission, and manipulation facilities than paper documents [84, 150]. Unfortunately, only 2 to 10% of information exists in digital, machine-readable form [137]. However, when a document management system contains thousands or millions of pages, the main issue is the retrieval of those and only those pages that are relevant to the users' immediate needs [132]. The average executive in USA spends grand total of about four weeks per year waiting for documents to be located [84]. "If the system holds back crucial information, then, in effect, it lies. [132]" This problem is studied by a variety of text retrieval related disciplines including library science and information retrieval (IR) developed from the Boolean query paradigm. Also recent developments in related areas -- such as natural language processing (NLP), document architectures, and hypertext -- provide complementary methods.

The combinations of knowledge representations (KR) and text can be expected to combine the clarity of formal knowledge representations with the flexibility and ease of natural language. From KR point of view, it is natural to supplement operational systems with textual information or even with natural language generation facilities to explain their reasoning. From the documentation point of view different formal structures have been used to aid the users to find the required piece of textual knowledge without knowing or memorising its location in the documentation or the specific terminology used in it. Domain-specific subject classifications and thesauri, hypertext and other document structures, as well as knowledge representations generated from text using natural language processing techniques can be used for this purpose. In these approache,s KR has essentially a descriptive function in contrast to its operational nature in pure KR systems.

The problems of integrating and co-ordinating the use of KR and text can be seen as text retrieval problems of defining the relation of the representations and the central concepts or "aboutness" of the texts. Although the inherent nature of this problem is closely related with the extremely difficult natural language understanding problems, the partial solutions are useful for text retrieval purposes. In the text retrieval approach, the natural language text serves the primary role of a knowledge transferring medium whereas the role of KR is solely to aid as (a part of) a text retrieval methodology supplementing the knowledge transfer by text. With the facilities provided by contemporary computer technology, the role of KR supporting knowledge transfer by text does not have to be so limited. In the somewhat broader standpoint adopted in this work, the primary role of KR is still to serve as a classification structure used for text retrieval, "a small amount modelling effort, ..., produces a significant improvement in our ability to extract pertinent information [60]."

Secondly, KR views the world from the "common knowledge" point of view. It includes concepts relevant for an organisation and gives names to them. It reflects relationships that exist between concepts, it must enable users to navigate it, and the criteria for classification in it must be explicit [25, 241]. "A topic index like this becomes an organisational asset, reflecting the basic categories in which the organisation view the world. [241]" This kind of shared information will serve as a distributed collective memory of large organisations -- a "shared mental map [204, p. 12]" or images of the world. This way, communication between members of the organisation can be based on a common ground, which reduces misunderstandings and improves communication through documentation and other language-based media. Thus the models can also be used as such -- for learning the common view of the organisation and for creating a context for the knowledge contained in documents -- not only as topic index for classifying and indexing.

The more emphasis we give to the representations of domain knowledge, the more critical is the ability to construct high quality domain models efficiently. Unfortunately, the traditional text retrieval knowledge representations have typically clear limitations for these purposes, or methods used for their construction are highly elaborate. The subject classifications and subject headings lack expressive power, for example, to represent the various relations between concepts. The use of an advanced thesaurus instead of subject headings gives more expressive power, but requires great effort in thesaurus construction and maintenance. The knowledge representation methods developed in the KR research have some potential advantages in representing domain knowledge, but they also tend to suffer from the similar problem, known in this discipline as the knowledge acquisition bottle-neck.

This dissertation is mostly based on the work performed in close connection with the SIMPR project, although contains parts from other related work of the author, especially in the fields of model-based diagnostic advisory systems and hypertext. SIMPR (Structured Information Management: Processing and Retrieval [169]) was the project number 2083 of the second phase of European Strategic Program in Information Technology (ESPRIT II). The SIMPR project was a 3.5 year (64 person-year) project running from 1989 to 1992. The 9 groups from 6 European countries included both academic and industrial organisations: CRI A/S (Coordinator, Denmark), Catholic University of Portugal, Dublin City University (Ireland), Nokia Research Center (Finland), Cap Gemini Innovation (The Netherlands), TNO (The Netherlands), University College Dublin (Ireland), University of Helsinki (Finland), and University of Strathclyde (United Kingdom). The project developed methodologies for the management of large text databases and for the processing and retrieval of technical documentation. The approach used combines information retrieval (IR) and natural language processing (NLP) methods. The main areas of work were semiautomatic indexing based on NLP, classification based on faceted subject classification or domain models, database management, and text retrieval [72, 97, 107, 108, 196, 198,

200, 201, 228, 232]. The DTM domain modelling system described in this dissertation is a result of the work carried out by a research group at Nokia Research Center in which the author worked as a member and later on also as the leader of the group.

1.2 Assumptions and Objectives

The context of this work is in using a combination of formal KR and text to *transfer technical knowledge* about large products and systems. By this we mean transferring system-dependent information and knowledge -- here referred as *domain knowledge* -- from system designers to the users of the systems. This knowledge is used for operating the systems, for solving problems, and for updating users mental image of the systems, necessary in order to the user to understand systems' behaviour. *Domain models* refer to formal representations of the domain knowledge describing central conceptual structures of a domain. This work focuses on domain models supplementing text retrieval in transferring technical knowledge, and especially on the methods for constructing these models (such as various thesaurus construction approaches described later on, pp. 18-20).

As an underlying assumption, natural language is used as the primary medium for domain knowledge transfer. This is due to the pains taking knowledge acquisition process needed for construction of KR and the ease of human interaction with natural language. The integration of KR and text is mostly treated as a text retrieval problem. Many of the problems of contemporary text retrieval systems are assumed to be based on the richness and inherent properties of natural language [180, 239, 247], on their poor capabilities to represent common conceptual knowledge of the domain to users [25, 241], and on the gap between the methods used for solving these two problems. With respect to the approaches concentrating on text retrieval methodologies alone, this work takes a somewhat more knowledge-oriented and domain-centered approach in supplementing the text with KR.

The main objectives of this work are:

1. To analyse representations for domain knowledge in text retrieval.
   This includes the analysis of the existing general purpose text retrieval approaches, especially the use of knowledge representations for representing domain knowledge, and the relation between these representations and text.
2. To analyse the special features of technical documentation and domain modelling.
   This includes the analysis of the special features of technical domains, documentation, and text retrieval. Special emphasis is on analysing the applicability of the general purpose means
   - to represent domain knowledge for text retrieval purposes,
   - to construct domain models, and
   - to transfer domain knowledge to users for supplementing technical documentation retrieval.

3. To design and evaluate a domain modelling method applicable for technical domains. This includes
   - finding a modelling approach matching the special requirements of technical domain modelling,
   - stating the requirements for domain knowledge representations supporting the approach,
   - designing, implementing, and evaluating representations and modelling methods satisfying the requirements, and
   - analysing the influences of the methods.

1.3 Scope

This work is based on the development in knowledge representation in recent years, and the readers are presumed to have some back-ground information in this area. In this work, the problems and advantages of using KR to supplement text are reduced to combining the results of KR and text retrieval disciplines, i.e., how to acquire and represent knowledge and how to integrate KR and various text retrieval methods to enable the integration of KR and text. These issues are studied from one point of view by knowledge representation (KR) and data modelling approaches, and from the other by a variety of disciplines that focus on or can be applied to text retrieval. The basis of the text retrieval view relies on library and information science and on the IR paradigm whereas the form of the analysis of representations is derived mostly from NLP, document structure, and hypertext approaches. Only reference is made to some of the related practices, such as text understanding, intelligent databases, and storage techniques including object-oriented databases. Others are excluded, for example, user modelling and computer-supported co-operative work.

The focus in this work is on corporate documentation systems in technical domains although major parts of the work are also generally applicable, to some extent even to online databases. In this work, the focus in text retrieval systems is on the representation, especially on representations for conceptual knowledge, that would aid in the text retrieval process, not on knowledge-based retrieval, question answering, or other related issues addressed by some knowledge-based and text retrieval systems. The view adopted here emphasises the declarative nature of the representations as opposed to the procedural features of the text retrieval process.

1.4 Contents of the Dissertation

An early version of this work has been published by the author as an external SIMPR report [231], although major parts of the work has been performed outside the project. The dissertation is organised as follows. After this introductory Chapter 1, the Chapter 2 provides an overview to text processing and retrieval. The approach adopted in this work is first, to analyse the existing, diverse text retrieval approaches. For this purpose Chapter 3

devises a framework observing the approaches from a common point of view, from the levels of representations used in them. The approaches described and analysed in Chapter 4 include conventional library methods and IR approaches, NLP-based text retrieval, and structure-oriented text retrieval including hypertext. This chapter presents also a hypertext generation method for diagnostic hypertext advisory systems. The view of text retrieval created in this and the previous chapter is used as a basis for the analysis of special features of technical documentation and domain modelling later on. The framework and the analysis following it are original work by the author. The framework has been outlined earlier for the analysis of hypertext text retrieval [230]. Also the hypertext generation method has been described earlier [226, 227].

Chapter 5 focuses the attention on technical documentation. The role of technical documentation will be seen as a medium for transferring knowledge which is needed for solving problems for accomplishing the mission of an organisation. This chapter presents special features of technical documentation (reported also in [92, 211]), describes the problem-oriented nature of text retrieval in technical domains, introduces effort/result diagrams for analysing text retrieval performance (reported also earlier [229]), and discusses the economical criteria for use of a TRS in technical domains. Parts of the economical aspects have been outlined earlier [228].

Chapter 6 analyses special features of technical domain modelling and proposes a new principle as a guideline for this purpose. The contemporary modelling approaches are compared with the special requirements, and a specialised method for modelling technical domains for text retrieval purposes is outlined. The last sub-chapter summarises the requirements for knowledge representation models and lists the shortcomings of contemporary representations of KR in various text retrieval approaches.

Chapter 7 presents a domain models designed especially to meet the requirements of technical documentation retrieval called Domain and Task models or D&T models. It also illustrates text retrieval with the D&T models. This basic model has been designed and implemented jointly by our research group at NRC, within the scope of the SIMPR project. It has also been described by our research group [232, 233, 97] and in external SIMPR reports [95, 96, 183]. The author has had a major influence on the development of the basic model, e.g., by focusing the design on the requirements stated earlier. Apart from the joint design, the author presents alternative ways to use the D&T domain models. The further analysis and methodology development can be considered as the author's work, although they have been partially implemented by other members of the group and reported jointly in the SIMPR reports.

In Chapter 8, the author first compares quantitatively the use of D&T models with the use of hypertext and introduces the CIOS architecture adopted to text retrieval interfaces (reported also in [230]). The chapter presents DTM modelling methods following the modelling principle proposed for technical domain modelling in Chapter 6 and gives

guidelines for their use. Parts of the ideas have been outlined in earlier papers [228, 233]. The techniques have been implemented and reported jointly by the research group [93, 94].

Chapter 9 summarises the exploitation of the DTM domain modelling system as a part of TeleSIMPR pilot system in a technical environment, describes the domain modelling at the pilot site, and outlines the results gained from the evaluation. The author analyses the results of the pilot experiment, describes the problems of the approach, and compares the approach with other related work.

Finally Chapters 10, 11, and 12 summarise the results, draw the conclusions, and outline suggestions for future research.

2 TEXT PROCESSING AND RETRIEVAL

2.1 Central Concepts

Salton and McGill define the concept *information retrieval* (IR) as follows [190]: "Information retrieval is concerned with the representation, storage, organisation, and accessing of information items". In practice this expression has almost been a synonym for keyword-based querying of bibliographic databases. This work prefers the use of another common term, *text retrieval* (TR). Here text retrieval is viewed as a process whose goal is to satisfy users' information needs by retrieving texts describing the required concepts or topics. The role of a *text retrieval system* (TRS) is to aid the users in obtaining texts containing the required information. In Soergel's terminology [204] a text retrieval system is viewed more as an information system rather than as an information storage and retrieval (ISAR) component of an information system. This means especially that the user input is not supposed to be a query statement to an ISAR system, but rather, a TRS is supposed to have a component for acquisition of the needs of specific users approaching the system with problems and resulting information needs. Although the connotation of term *information retrieval system* (IRS) is closer to an ISAR component than an information system, it is used here as a synonym of TRS. In general, the view adopted here is based on authors previous work in knowledge representation and thus closer to the cognitive view (see f.ex., [98]) than the narrow IR approach.

In this work the term *text* means a piece of information that can be retrieved by the text retrieval system and is relevant to be treated as a separately manipulated entity, for example a document, a chapter of a document, a paragraph, or a page of a hypertext system. Similarly a surrogate, i.e., a record of a secondary (bibliographic) database representing an original text, can be considered as a text, although online databases [50] are out of the focus of this work. The contents of a text may include natural language, structured text in form of lists and tables, and also graphics and figures. Most of the contemporary methods for figure and graphics retrieval reduce it to text retrieval by using texts as content descriptors. Thus graphics recognition methods are not considered to be relevant for this work. The terms *document, information unit,* and *chunk* are often used in the same meaning as term text here. Here the term document is used for a typical technical or commercial report containing few or tens of pages each containing a couple of texts. The texts are assumed to be included in a *documentation*, i.e., a fairly large body of texts describing a reasonably uniform and somewhat restricted area of interests, such as a technical domain.

The identification of relevant texts may be based on a variety of methods including utilisation of the *structure* or *formal properties* of the texts, such as the relations and references between text segments or properties, such as the author, the language, the name of the project that has produced the text or even the type of the text, such as a report, a

letter, or an invoice. Many contemporary systems use *content identifiers* or *content representatives* that are supposed to represent the main content of the texts with a reasonable accuracy. Keywords, single or multiple-word index terms, and other descriptors are used for this purpose. If all or most of the words of a text are used as keywords, the system is called a *full-text retrieval* system. The texts may also be *classified* according to their content to one or many categories called *subject classifications, subject headings,* or *clusters*.

Before a text can be retrieved with a text retrieval system, it typically goes through a text processing phase. This can include extraction of text structures as well as indexing, i.e., generation of content identifiers, or classification of a text requiring a substantial text analysis effort. Document abstracting entails creation of synopses of lengthy documents highlighting key concepts and significant data contained in the documents. The author or the librarians may perform parts of the processing phase manually, for example, by classifying the texts under certain subject headings. Some techniques rely on computerised methods, such as statistical methods to index or cluster texts based on word frequencies in a text. There are also various preparations that must be performed before the text processing can take place; build up of database structures and text format converters, tailoring needed for languages used, building of domain-dependent vocabulary (lexicons), thesaurus describing relations of words, word frequency statistics, concept hierarchies, subject headings, etc.

In a *text retrieval* process the user typically represents the request as a *query* whose formalism is defined by the system. The text retrieval system then *searches* for texts containing identifiers similar to the ones in the query and returns the ones matching the query with adequate degree of similarity. The user may also *browse* or *navigate* a subject classification scheme, a thesaurus, or some other domain model to find the texts classified under the interesting topic. The texts themselves can also be browsed by browsing the hierarchical chapter sub-chapter structures, by following *hypertext links* between the texts, or by browsing through abstracts.

The *effectiveness* of query oriented text retrieval[1] is usually measured by two figures, *precision* and *recall*. Precision is the proportion of retrieved texts that are relevant and recall is the proportion of relevant texts that are retrieved. These two measures are inter-related in following way; broadening a query on one hand increases the number of relevant texts retrieved and thus the recall, but on the other, decreases the precision since more irrelevant texts are also retrieved.

$$\text{Precision} = \frac{\text{Relevant retrieved texts}}{\text{All texts retrieved}} \qquad (1)$$

[1] I.e., the effectiveness of an ISAR component. Further discussion about the measurement can be found in following chapters.

$$\text{Recall} = \frac{\text{Relevant retrieved texts}}{\text{All relevant texts}} \qquad (2)$$

2.2 Historical Background

As the amount of primary textual information, books and articles, grew along the centuries, the use of referential information became necessary. These secondary sources were used to refer to the main contents of the primary material without forcing to go them through. The early techniques used were abstracting, indexing, and use of subject classifications. The availability of secret technical reports after World War Two was a major impact to development of text retrieval systems. The two conventional types of index used those days were card catalogues and annual accumulation of an abstract journal or some other printed media. The contemporary major indexing techniques were classification systems in Europe and alphabetical subject headings in America. [34]

The use of index terms with a thesaurus, subject headings, and subject classifications has been fairly similar [148]. Subject classification systems, such as LCSH and MeSH[2], can be defined as controlled vocabularies of index terms with *thesaurus* relations between the terms. The relations of subject headings in LCSH and other similar system can be mapped to the typical thesaurus relations: *see also* reference is partially covered by *related term* and partially by the *broader term* / *narrower term* relations, the *see* reference is directly equivalent to the *use* reference in thesaurus. I.e., these approaches represent mainly similar relations at different levels of specifity, hierarchical categorisation relations, synonymy relations, and other references to closely related terms. In case of controlled vocabularies the non-accepted synonyms of an accepted index term use the *see* or *use* references to the preferred index term. Even though, the classification schemes emphasise more the taxonomic organisation of the subjects whereas the index terms are more oriented towards "ad hoc" description of text content with loose interrelationships, the thesauri approaches being the most specific in describing the relations.

During the early 50s there were many attempts to depart from the conventional system such as the development of facet classification systems in England trying to break away from the enumerative or hierarchical classification and the work in America on Zatocoding and semantic factoring. The most successful of these was the work of Mortimer Taube on the Uniterm System. According to his observations there were only some 7000 different words in the 40 000 subject headings of a government library. These individual words could be used as index terms which would be co-ordinated at the search stage. [34]

In the first Cranfield tests the Universal Decimal Classification, a conventional alphabetical subject index, a purposely devised schedule of a facet classification, and the Uniterm System of Co-ordinate Indexing were compared. The results showed that each of

[2] LCSH = Library of Congress Subject Headings,(USA) MeSH = Medical Subject Headings by National Library of Medicine.

the systems achieved approximately 74 - 82% effectiveness in retrieving the required single paper considered to be the answer for the question (the Uniterm system showing a slight advance and the faceted classification having the lowest rank). The analysis of the failures indicated mostly human errors and lead to a presumption that the particular system used appeared to have no significant effect on performance. [34, 35]

The second Cranfield tests took the view that all index languages were amalgams of recall and precision devices (i.e., methods get all relevant texts and exclude irrelevant ones) and the objective would be to define the effect of each of these devices. The indexing was based on concepts expressed usually by two or three words. 29 indexing languages were devised based on basic natural language, concepts, and controlled language terms. Various recall and precision devices were created using word forms, synonyms, quasi-synonyms, and three stages of hierarchical reduction of the natural language terms. Concerning the specifity of index terms the results showed that starting from single natural language terms the additional recall and precision devices did not improve the performance (with the exceptions of confounding word forms and true synonyms[3]). [34, 35]

With respect to exhaustivity the results of the second Cranfield tests showed 33 terms to be the optimal size of index terms to be used for the test documents. The use of abstracts and further, the use of full-text searching would degrade performance in the test environment [35, 34]. Together these two results suggested that text indexing should be based on a limited number of index terms (multi-word content representatives) whose constituents, individual words, are used as the set of text content representatives during the retrieval.

2.3 Early Retrieval Systems

The result of the Cranfield tests concerning the specifity has been adopted gradually and the early computerised text retrieval methods preferred use of inverted files for single words. Concerning the exhaustivity later computerised methods tend to favour full-text indexing instead of extraction of index terms or abstracts, i.e., (almost) all single words occurring in a text are used as the index terms instead of only the words in manually assigned index terms or words of the abstract.

The retrieval effectiveness in full-text search is not as good as with the use of index terms [34]. According to the Cranfield tests the use of a limited set of index terms improves precision; terms that are occasionally used in a text describing mainly some other topic do not cause irrelevant hits as in full-text retrieval systems. From the storage space point of view the use of index terms is very advantageous. A text is represented with only a few classification codes or index terms, which reduces the need for comparisons during the retrieval phase and enables faster response to user requests. Manually assigned index terms

[3] The use of true synonyms is generally accepted and supported statistically. See f.ex. [122].

may also help the users to find texts that do not explicitly contain the requested term(s) that were assigned to them during the indexing phase based on their content.

The advantages making full-text search popular are mainly economical. Abstraction process, manual extraction of index terms, as well as maintenance of controlled index terminology are very costly. Thus, for example, abstraction approach has been replaced by full-text retrieval systems in most domains except certain industries [62]. Texts may also contain terms that the users may use in their requests but are missing from the keyword list. The texts containing these missing keywords are retrieved by a full-text search and occasionally they are relevant for the request.

With respect to implementation and retrieval, the assignment of single word index terms and full-text search are very similar. The major difference is in the exhaustivity of the indexed material. The index terms are typically restricted to a pre-defined set of terms with more or less clear semantic definitions, which helps to handle problems caused by inflections or use of synonym terms. In practice, the situation is not quite as good as it should be, although use of index terms improves retrieval effectiveness [34]. In both cases the implementations are based on efficient access to the relationships between texts and terms. From the point of view of texts each text contains a set of terms that may be represented as a term vector describing the absence or number of occurrences of a term in a text. From the point of view of a TRS, the relationships are often implemented as inverted file(s), where each index term or word in an alphabetical list refers to all its occurrences, i.e., realises the mapping: term -> occurrences in texts. This enables fast access times but requires 30-150% additional storage space compared to the documents [219]. The implementation of inverted files uses typically efficient access structures, such as B-trees or hashing. With small amounts of text the texts in a full-text retrieval system can be searched through to find the occurrences of a term. The specialised text scanning or more elaborate text skimming methods [86] utilise available processing resources and do not need additional storage space for intermediate inverted files. The speed of scanning can be improved by compression, by sophisticated pattern matching algorithms, by signature techniques as well as with specialised hardware.

An average information retrieval system contains thousands of pages of texts or millions of abstracts in a reference database. With a Boolean search interface the user searches for entries containing a certain word or combination of words. The requests are represented as queries where the words are combined with Boolean operations AND, OR, and NOT. For example, the query

```
car AND (motor OR engine)
```

accepts all texts that contain the term "car" and either the term "motor" or the term "engine". Parentheses are used here to define the order in which the operators are applied. Measured

retrieval effectiveness values for well defined Boolean queries are typically something like 30% recall and 50% precision, but quite often even less. [35, 69, 186, 217]

2.4 Thesaurus Construction

Thesauri and subject classifications have been the major knowledge representations in early text retrieval system. From the point of view of this work the thesaurus construction process is especially interesting and is described shortly here for further use in this work. The following overview is adopted mainly from Lancaster [124, pp. 21-48]. Creation of classification schema is not treated separately, since it is included in the thesaurus construction process; "It is impossible to arrive at an effective thesaurus structure without first constructing a classification scheme [124, p. 36]."

The construction of a new thesaurus is not always necessary. In many cases it is advisable to check the possibility that some existing thesaurus could be adopted to meet local needs. Another possibility would be to construct a microthesaurus to fit within the hierarchical structure of some more general thesaurus. Such approaches would probably be less expensive than building a completely new tool. [124, p. 21]

The construction of thesauri consists of two major phases, *the collection of the raw material* (i.e., terms) and the *organisation of this raw material* to form a consistent and effective tool for information retrieval. There are three major approaches to gathering the raw material; a *top down* approach and two *bottom up* approaches. The top down approach starts from the "top" term (summum genus) of a particular hierarchy through the various levels of subdivisions are established through intellectual effort, by subject experts. The top down approach has problems in identifying all the subdivisions and even all the major classes that would be needed in a field. The bottom up approaches first gathers the raw material and builds the hierarchies empirically from the collected terms. The bottom up approach tends to be more reliable than the top down approach. [124, pp. 21-23]

The first bottom up approach is guided by *literary warrant* principle. Literary warrant was first considered for bibliographic classifications. It states that the classification scheme should be empirically derived from the published literature. This principle is also applicable to thesaurus construction; An index term is justified only if it is known to occur in the literature of the subject field (often enough to be considered significant and useful for retrieval purposes). In this approach the terms are gathered either manually or with the aid of computers, from dictionaries, clossaries, encyclopaedias, handbooks, and comprehensive textbooks. The major problem of this approach is the human effort needed for the elaborate creation of thesaurus relations (BT/NT, RT, synonyms). Thus it is impractical[4] in a very large subject field, and viable for a small scale thesaurus. This process can be aided by term

[4] Although not impossible, if sufficient resources can be allocated, as for construction of a general technical or medical thesaurus.

occurrence statistics and term co-occurrence statistics that have potential value in identifying useful relationships among the terms. [124, pp. 22-26]

The other major bottom up approach to thesaurus construction is guided by the principle of *user warrant*. It states that a "term is justified for inclusion in an index only if it is of interest to the users of the information service." This information can be achieved by recording past information requests, by consulting other information centers covering the same subject areas, by interviewing the potential users, and by using mailed questionnaires to a random sample of potential users. The results are then processed as with the former approach. Ideally, both bottom up approaches should be used together. As a thesaurus is a dynamic tool, it is never likely to be complete. Thus the organisation of terms into a coherent thesaurus structure has to be started when the gathering of the terms is still going on. [124, pp. 26-28]

The second major phase in construction of thesauri is the *organisation of the raw material* to a coherent and cohesive structure for effective information retrieval. This includes defining two kinds of relationships between the terms, *hierarchical relationship* and *associative relationship*.

The *hierarchical relationship* is accomplished in two phases. First, the terminology is divided into a series of aspects or facets. F.ex., library science terms can be divided into facets, such as "Kinds of Libraries", "Library Materials", "Library Services", and "Other". In the second phase the terms in each facet are organised into broader term / narrower term (BT/NT) hierarchies. Lancaster states also that it is impossible to arrive at an effective thesaurus structure without first constructing a classification scheme. [124, pp. 35-36]

In general, the hierarchical BT/NT relationship should always be one of genus/species; i.e., the species term must represent a "kind of" the genus term. If it does not, it is probably not a legitimate NT. This thing/kind relationship (the true BT/NT relationship) should not be confused with the thing/application or thing/derivative relationship. According to Lancaster the whole/part (partitive) relationship is -- in most circumstances -- also not a legitimate BT/NT relationship (cp. ISO standard [76]). If different principles of division are used (like shape, material, and application) it is convenient to arrange NTs by principle of division. In general, the "hierarchical relationship is relatively clear-cut, and rather precise guidelines can be formulated to ensure that the BT/NT relationship is consistently applied. The association relationship is much less clear-cut". [124, pp. 39-44]

The *associative relationship* is a non-hierarchical relationship represented in the thesaurus by RT, meaning "related term". It is syntagmatic or a posteriori and it is not possible to establish precise rules for it. The one thing definite about it is that it should not be used to link terms appearing in the same hierarchy. Otherwise any two terms whose meanings are related are possible candidates for linking by the RT reference. This means that there are various RT relationships types, such as thing - application, cause - effect, thing - property, material - product, complementary activities, opposites, activity - property,

activity - agent, activity - product, and thing - parts (see also [76]). The best formulation for an acid test is: "Is it likely that someone seeking information indexed under term A might also be interested in information indexed under term B?" If the answer is yes, the A and B should be linked by the RT reference -- providing that the relationship does not already exist via the BT/NT hierarchy. [124, pp. 45-48]

Some authors have suggested the possibility of arriving at useful RTs through a type of associative game resembling the "free association" used in certain types of psychological testing, but such a procedure seems unnecessary and artificial. Similarly, generation of tables of term co-occurrences from an appropriate database can indicate which terms are "related" in an associative sense. If this is not possible, the thesaurus builder can rely on common sense and on his knowledge of the subject matter. [124, p. 46]

With respect to the lists of subject headings -- as traditionally used in libraries -- the (more or less) clear distinction between the BT/NT and the RT relationships in a thesaurus demonstrates definite advantages. In the list of subject headings, both relationships are taken care of by a single reference "see also". Moreover, in conventional subject heading practice, the reference is made from the general to the specific but rarely in the opposite direction. Unlike a properly constructed thesaurus, the list of subject headings is not a perfect hierarchical classification, and one cannot automatically derive such a classification from a list of subject headings. [124, p. 48]

2.5 Text retrieval Problems

The richness of natural language causes many of the text retrieval problems [180, 239, 247]. For example, suffixes and affixes of terms and use of synonyms may cause a mismatch between a query and a text decreasing the recall of a query; existence of terms in different meanings (i.e., polysemy) or existence of related query terms as non-related words of a text may cause irrelevant texts to be retrieved decreasing the precision of a query. Also use of acronyms and anaphoras, typically pronouns, causes misses when certain words are searched. Extra hits are caused by use of analogy or metaphorical expressions. The inability to recognise structures containing many words causes problems when trying to find texts containing several words, such as parts of a compound word or an expression. The use of more general (broader) expressions as well as use of more specific expressions causes also problems with text search paradigm. On one hand the texts that should be found are not found and on the other, the searches made using general terms typically retrieve lots of unnecessary texts. From this point of view the text retrieval would be much easier if the natural language would be a "notational system" [75] characterised by semantic disjointness, i.e., two terms may not refer to the same meaning.[5] Word meanings do also

[5] This excludes f.ex. use of terms "doctor" and "woman" in the same language since there exists also female doctors. See pages 149-152, 178-179, and 226 [75].

shift in unpredictable ways based on the social and physical environment [44]. The use of a thesaurus to describe these relations improves typically the recall remarkably without sacrificing too much precision [17], but the task for building the thesaurus is domain-dependent and thus impractical for unrestricted subject domains [187].

The string orientation in general is a source of problems in conventional IR and should be aided by representations of the central concepts. One cannot request information unless one is able to name the thing that is needed [88, 220]. Research suggests that the mismatch between query and indexing vocabularies may be a major cause of poor recall [65]. The searching methodology is heavily based on the matching of strings in a form or another. The use of subject classifications or assignment of index terms is useful but does not fully support string independent information retrieval. The conventional methods for the construction, tailoring, and maintenance of classifications and index term systems are highly elaborate. Retrieval based on concepts and relations between them would be needed [10, 177]; including both hierarchical relations to support decisions about conceptual similarity and non-hierarchical relations for describing other relations [146, 177]. These kinds of representations for the central concepts of the subject domain should be efficient to build and to use for text indexing and classification.

The indexing task has been carried out manually, by librarians. As the amount of information grows the effort needed for indexing, classifying, and/or abstracting texts becomes unmanageable [62] . Also the variance of index terms and the problems of users to adopt to the selection of keywords and classifications used in an IR system enforces unification of methods but causes problems when changing the schemes [34, 155]. Extraction of the central content and concepts, i.e., indexing and classification, is knowledge intensive and the results vary according to the persons. As summarised from various sources [34], "... if two competent individuals prepare a thesaurus on a given subject, only 60% of the terms may be in common. If they index a document, only 30% of the terms may be assigned by both indexers.[6] If they carry out a given search, only 40% of the citations will be retrieved in common, and if they decide the relevance on a given set of documents, there may be only 60% agreement."

The practical aspects of conventional IR systems cause problems [69]. Each of the systems have different query languages and notations for operations. The facilities provided for the user vary from system to system and require more learning than is usually expected. The users have also difficulties in realising the size and contents of the database, which is invisible behind the query language interface. The users are not able to estimate the recall based on the responses of the system and are thus uncertain whether they should stop the useless retrieval session or redefine the query and try again.

---

[6] According to Furnas [65] the probability of two people applying the same term to a recipe was 18%.

The Cranfield tests showed the direction of single term indexing to be the method producing best results. The Boolean technique propagated by Taube and others aimed at improving index searches performed manually. Although the further developments of statistical and vector-space models display great improvements in performance compared to the original Boolean techniques, the full utilisation of high-speed computers and new methodologies have still neither reached the general acceptance of the IR community [81] nor the mainstream markets. As stated by Cleverdon [34] "At no stage in this development was the improvement of sufficient significance to justify a rethink of what was being done, but it is ironic that we continue to use such a search technique when computers are now doing in a fraction of a second what would have taken hours to do when the Boolean search technique was devised".

2.6 Artificial Intelligence

Several of the methods developed under title Artificial Intelligence (AI [194]) have been applied to text retrieval problems, especially natural language processing (NLP), knowledge representation (KR), expert systems (ES), object-oriented programming (OOP), and graphical user interfaces (GUI).

One of the central issues of artificial intelligence has been the representation of knowledge in a form that can be manipulated (efficiently) both by humans and by computers.[7] The early knowledge representation and reasoning systems were integral parts of individual applications and were tailored to meet the needs of each specific case or problem domain. They were often based on some specific representation such as formal logic, facts and rules, semantic nets, frames, or objects [23, 39, 82, 151, 210]. Later on, growing numbers of researchers have been working on more general representation schemes that can be utilised in a variety of applications over broader domains. All of the contemporary knowledge representation systems are essentially hybrid [197].

From the point of view of this work the most interesting issues in knowledge representation include representations enabling the users to define their information need without explicitly naming it. This means that representations should be somewhat string-independent (i.e., natural language independent) and rely more on alternative methods to represent knowledge. The long tradition of classification systems has certainly something to offer for this purpose and the later knowledge representation formalisms can also be utilised. Hierarchical knowledge has repeatedly been shown to be fundamental to the decision-making process from several points of view [179, 153, 205, 249]. For both abstract and concrete concepts, hierarchical categories are natural for people and for their decisions of conceptual similarity [1, 177, 179]. It is also generally claimed that the concepts (or other representations used for representing documents) should have more than hierarchical relations among them [10, 177, 220]. From this classification-oriented point of view the

[7] Without forgetting the cognitive aspects, see f.ex. [36].

deductive capabilities of KR are less interesting. Instead, the declarative classification and information modelling features found, for example, in conceptual information systems [106, 149] are interesting[8] as well as the means for effective man-machine interaction, knowledge communication, and visualisation via graphical user-interfaces [140, 212, 218, 243].

The major knowledge representation issue this thesis addresses is the knowledge acquisition needed for building domain descriptions [116, 134, 163]. In the context of text retrieval systems the knowledge acquisition bottle-neck can be found in the creation of the world or domain models, such as the classification taxonomies or thesauri (see page 18).

2.7 Natural Language Processing

Many problems of text retrieval are caused by the richness of natural language, as described above. These issues are traditionally addressed by linguistics and the results of recent computational linguistics are most promising also for text retrieval purposes. Traditionally the complex field of language is roughly divided into a set of interconnected disciplines that are treated during NLP as a set of rather independent layers, i.e., processed one level after another. Although this approach is rather simplified and also criticised deeply [59], it serves well as the outline of NLP for the purposes of this work. The levels are:

- phonology
- morphology
- syntax
- semantics
- pragmatics

Phonology considers the formation and combination of phonemes, how words are realised as sounds. Hence it is not in our interests as far as text retrieval systems (with no speech interfaces) are considered.

Morphology deals with the forms of individual words, how words are constructed out of more basic meaning units called morphemes. The word form determines to some extent the type and the function of the individual words. A word can be decomposed into word stem and affixes, (i.e., prefixes and suffixes). One word stem may have several inflected forms. In English texts there are some 75 prefixes and 250 suffixes. In languages with rich morphological structures the number of possible word forms may be huge; for example in Finnish a verb may have as many as 50 000 inflected forms. The word stems are typically stored in a dictionary or in a lexicon. An entry in a lexicon used for morphological analysis contains a word stem or a baseform and additional morphological information such as the lexical word category or categories for each interpretation of the stem. Each entry may also contain more or less additional information such as syntactic and semantic descriptions for

[8] Although not as relevant from the point of view of this work as the term "terminological reasoning" used in this area may sound.

each word sense and thesaurus, such as references to other lexicon entries. Some lexicons include also multi-word phrases. [5, 107, 108]

Syntax deals with the structural properties of natural language, i.e., formation of sentences from phrases and individual word forms. Various methods are used to describe the syntax of well-formed text statements, e.g. grammars based on theory of formal languages, automata, constraints, and statistical methods. The most popular method to represent the structure of an individual sentence has been a tree structure with a sentence label S in the root and the individual words in the leafs. If the syntax is described by a set of syntactical rules, each of the rules describes the expansion of a leaf in the tree to a new level of nodes. [5, 32, 70, 105, 107]

Semantics concerns the meaning of expressions, what words mean and how their meanings combine in meaning of sentences. Pragmatics concerns the use of expressions or sentences in different contexts and their interpretation in different contexts. When NLP is used for text analysis, the semantic processing is typically applied to the structures extracted by syntactical analysis,[9] i.e., the semantic component "interprets" syntactical structures by assigning them context-independent truth conditions. After this a pragmatic component fixes up this literal interpretation and takes the context into account. This includes use of world knowledge, the general knowledge about the structure of the world that language users must have in order to communicate with other human beings. [5, 70, 146]

In general, the problems related with pragmatics and world knowledge are inherently extremely difficult or even impossible to solve. Consider, for example, the problems with discourse semantics in story understanding, where the human readers ability to recall references to the previous sentences, and chapters as well as references to ancient history or local politic events should be automated by NLP. From the point of view of hermeneutics the possibility of formalising mental processes behind understanding and interpreting text is still an open issue [194, pp. 362-376]. Thus the language related problems of text retrieval seem to keep a complete solution to text retrieval problems out of our reach.

2.8 Document Structures and Hypertext

Where NLP treats the texts as streams of characters, the document architecture approach is interested in explicitly marked, i.e., tagged, structures of texts such as the separation of a documentation to individually manipulated documents and the partition of a document to headings, paragraphs, tables, etc. This kind of explicitly marked structures are simple to utilise for information retrieval purposes. For example, in several text processing systems the user marks the phrases used as the index phrases in the back-of-the-book index, where the phrases and the page numbers of their occurrences are listed. This information can also be used for phrase indexing of text databases as well.

---

[9] Although also semantics-first approaches exists [135].

The term hypertext [13, 38, 157] refers to a way to organise the documents or documentation into small chunks of texts called *nodes* (or cards or pages) that can be read in a variety of orders by following *links* between the nodes. The linear structure of a traditional document can be represented as a chain of nodes starting from the first page of the document and ending to the last page. The links may refer to texts describing persons or subjects referred to in the texts as well as to documents referred to in the text. In general, the links between nodes are based on authors' associations between the nodes and the hypertext networks are somewhat analogous to semantic networks containing concepts and links between them. Proponents of hypertext contend that non-linear information processing mirrors two natural patterns of human information processing -- association networks and hierarchies. One should still remember that hypertext is not a form of artificial intelligence, but is intended to augment human thinking by providing a dynamic platform for processing and presenting data [27].

Although a hypertext document can be represented as a bundle of note cards in a shoe box the advantages of hypertext are related to the computerised environments where the nodes and their relations are stored on a database and the contents of each node is represented as a separate window on the screen. The systems referred to as hypertext systems contain text and graphics. A computer system that is also able to represent time-dependent medium, such as voice and moving images (video), is called a *multimedia* system. A natural way to combine these separate medium is to represent them in a non-linear order, in form of *hypermedia* [38]. The typical way to use a hypertext system is to *browse* through the document by following links from one node to another by clicking emphasised words of the text or icons on the sides of the text that represent the links to other nodes. The purpose of browsing is to seek information without a priori knowledge what exactly should be searched. During this process the user may easily get lost in the hyperspace, i.e., lost awareness of the current position in the documentation and the aim of the process (also called *disorientation* [38]). When *navigating* a hypertext network the user knows the goal and tries to find a way to the target node. This process may be aided by a browser, a graphical display of the network displaying the path traversed so far and/or the next nodes linked to the current one. In addition, the hypertext systems may include facilities to *search* or *query* nodes containing a certain string in the header of the node, in the body of the node, or as an attribute value of the node.

The relation of traditional IR methodologies (searching) and hypertext systems has been analysed by Waterworth and Chinelli [237]. For their analysis they differentiate three dimensions on information exploration:

- Structural responsibility
  From users point of view navigation of hypertext is structured as from systems point of view it is unstructured. In case of queries the situation is reversed.

- Target orientation
  In (mediated[10]) browsing the user does not have a definite target in mind where as when querying it exists.
- Interaction method
  With descriptive interfaces the user describes what is wanted where as with referential forms of interaction the user selects or refers what is wanted.

Thus the effectiveness of navigation and browsing can be considered to improve by increasing the awareness of structure and by assisting the process of choosing between elements in the current structure, respectively [237]. Hypertext links may be used to promote structure such as paths and hierarchical links [54, 165]. Hierarchy provides an index structure and allows the user to explore categorical relationships. The hypermedia might also have the type of linking representative of concepts and their interrelations, similar to the semantic network approach to knowledge representation [237] (see also I^3R in [41] as well as [54, 68]).

In addition to the "getting lost in hyperspace" phenomena the creation of hypertext has been considered a major problem of hypertext. In [242] a hypertext database is created automatically from machine readable versions of ordinary texts based on a dictionary of term definitions and on cross-references of the documents. In Max [18] the hypertext interface dialogues are generated dynamically based on users tasks and a mathematical model -- as opposite to content(s) based access to pre-existing textual material. Citation links are also found to be useful in some approaches [41]. More advanced link creation methods under research include linkage of texts containing the same index term, utilisation of NLP [79, 123], and statistical similarity measures of text chunks for creation of links based on text similarity (c.f. classification by chunking or [188]). Linkage may also be based on learning of users behaviours as text relevant to two related queries are linked together or on index terms, that are in a causal relation in a text [43]. There exists also some commercial products that (semi-) automatically create a hypertext structure from linear text files (SmarText, Folio Views, InfoNavigator, and HyperBase [219]).

[10] Mediated browsing as opposed to navigational browsing, where the user navigates via the links to a pre-defined goal.

3 A PROPOSED FRAMEWORK FOR REPRESENTATIONS

3.1 Introduction

The problems of text retrieval systems are approached from several directions mainly from the point of view of library and information science and the mathematically oriented information retrieval (IR) community utilising statistical and vector based methods. Lately the methods developed in computational linguistics, cognitive sciences, and artificial intelligence research have been applied to text retrieval. Also the development of document architectures and hypertext have been used for this purpose.

In this work text retrieval is viewed as a process whose goal is to satisfy users' information needs by retrieving texts describing the required concepts or topics. The role of a text retrieval system is to aid the users in obtaining texts containing the required information. In this work the focus in text retrieval systems is on text representations and on representations for conceptual knowledge that would aid in this process.

The rationale of this chapter is to present a framework using which the diverse approaches can be analysed and compared. The framework selects one of several alternative views as the primary organisation, namely the representations used in the systems. The chosen point of view differentiates the representations based on their syntactic vs. semantic nature. The emphasis is on the declarative nature of the representations as opposite to procedural features of the text retrieval process. The next chapter classifies the representations used in alternative methods according to this framework and describes the approaches in general by characterising the typical features of the approaches and by listing references for further inspection of the diverse approaches.

3.2 Strategies

When starting a session with a TR system the users' interests are originally not represented in textual form, but rather exist in their minds merely as ideas about the required information. During the retrieval process the information needs are mapped to sets of texts that contain the needed information. These mappings may go through several intermediate phases controlled by a TR system or by the users, as in hypertext systems. The processes may utilise explicit representations of the information needs, representations for world or domain knowledge, as well as representations of the texts. The information needs may be mapped against taxonomies of the system represented using abstractions and knowledge representations, such as domain classifications and index terms. Before getting to the texts the processes are likely to go through a layer of text representations (content identifiers, formal properties of texts, or text structures) extracted from the texts. In the end of these processes the users have reached texts or sets of texts fulfilling their information needs. The

process may also end when the users become convinced that no such texts exist within the collection.

As described in previous chapters, many of the text retrieval problems are on one hand based on the inherent properties and inadequate or missing manipulation of natural language, and on the other the poor capabilities to represent and exchange common conceptual knowledge of the domain between the retrieval system and users. In contemporary text retrieval systems these problems are approached roughly from two directions. First, from the texts by extracting structural information and formal properties of texts or by using statistical and natural language processing (NLP) techniques to extract text representatives with several levels of NL processing. Secondly, from direction of knowledge representations (KR) to represent a common view of the world or a domain to the retrieval users and to the authors or indexers of the documentation. Depending on the level of text representations and the knowledge representations used their linkage may be problematic. Either this linkage is inadequate or its' creation requires substantial effort.

The first strategy to improve text retrieval is to process text, i.e., to cope with the inherent properties of texts with NLP techniques and statistical methods or extract structural features of the texts that can aid the retrieval process. With morphological and syntactical methods the systems can produce text representatives that avoid several of the problems associated with natural language text. With further processing the content of the texts can be represented by a language oriented knowledge representation formalism. This method avoids the drawbacks of many contemporary IR systems tackling with inherent features of natural language at the string matching level. It can be applied both to the processing of texts and to processing of natural language queries and supports thus searching based on comparison of text representatives. With no domain-dependent thesaurus or browsing aids the matching of representatives is dependent on the textual representations.

The other strategy starts from the direction of knowledge representation that is aimed towards interchange of common conceptual context between the system and its users, i.e., creators of the documentation, text indexers, and people retrieving the texts. The primary aim is to match the ideas and needs of the retrieval users to the ideas of the authors using knowledge representations of the system as a common language.[11] This language is less string-dependent than terms or phrases since the concepts can be described not only by their names but also by their properties, by their position in taxonomies, and by their relations to other concepts. The knowledge representations are used as a common ground for finding out what knowledge the user needs, what knowledge the world or domain model includes, and what knowledge is represented in the documentation. The relation of the last two is also valuable; the information about the amount of texts available about certain subject domain or the information that no relevant texts exists is useful. Although this knowledge-oriented

[11] This work is strictly focused on the representations of the concepts of the domain and excludes user modelling.

approach is most useful in text retrieval, the trade-of between elaborate knowledge representations and ease of use has to be considered and the connection of knowledge representations and the actual texts needs utilisation of other methods, such as NLP or conventional IR.

3.3 Levels of Representation

The framework presented here views text retrieval as a set of layers connecting the mental images of the users (i.e., ideas) with the texts describing them. Separate text retrieval approaches are viewed as different ways to utilise these layers in connecting the users' ideas with the texts. Figure 1 represents the layers of representation involved in text retrieval.

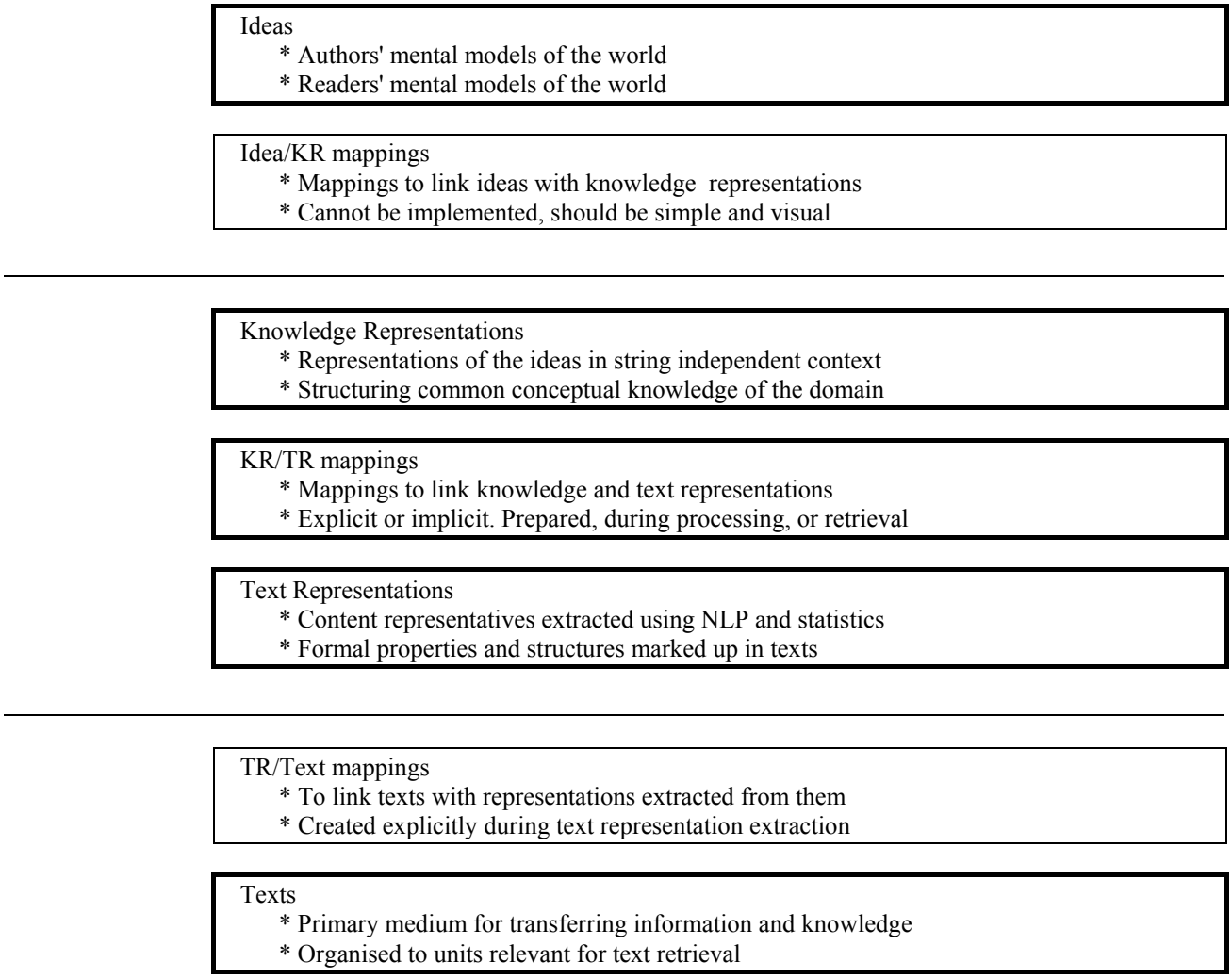

Figure 1. The layers of representations in text retrieval.

The idea layer contains the mental images and the concepts of the domain as they exist in the minds of the people. They reflect the world entities and human knowledge about them. On the opposite layer of the model there are the texts at the text level. A text is an authors' attempt to transfer sufficient portion of their ideas to the readers in form of natural

language, at the text layer. The role of natural language documentation is to serve as a flexible media for transferring information and knowledge of the idea layer from the authors to the readers.

3.3.2 Knowledge Representations

In this framework the knowledge representation (KR) level contains representations of the idea level entities, their properties, and their organisation. These representations of conceptual knowledge serve as a collection of common knowledge describing the structure and concepts of the world[12] or a domain from a commonly accepted point of view. The representations and their identity are as string independent as possible. The structures provide document readers with concepts organised to taxonomies and in relevant contexts. They enable the users to find the required concepts even when they are not able to request them by name. Thus the users are able to unify their information needs with the ideas of the authors by mapping them to the common concepts at the knowledge representation level -- the gap between readers' and authors' mental models of the subject information domain is reduced to the mapping between readers' mental models and the representations of the system's explicit model. With proper knowledge representations this mapping from ideas to knowledge representations should be as simple and clear as possible and should not incorporate problems, such as different language usage in different contexts, since all the representations are as string independent as possible and the identity of the concepts is based on the relations of the concepts. Of course, knowledge representation without text is in practice impossible, but misunderstandings about the meaning of the representations can be decreased with proper representations. Thus the mapping between a proper KR and the mental images of the users is supposed to be much easier than mapping information needs to unknown vocabulary or domain-specific terms at the text representation level.

In this framework the focus on the knowledge representations at the KR level is on representation of the conceptual structures and on interaction with the users. Thus the declarative aspects of KR are emphasised and the computational aspects are ignored within the framework, although not forbidden.[13] With respect to the general KR methodologies and frameworks the focus is on the domain layer of the KADS model [87] or on the task concept model according to Klinker's terminology [116], i.e., representations corresponding roughly to a logical data model of conventional systems. The representations may contain separate definitional components describing the overall structure and taxonomies of the world or domain, as well as components describing the individuals of the world or domain and their

---

[12] Due to the problems in world modelling the scope will, in practice, always be in a more or less limited domain, as discussed later on.

[13] Active intermediary functions and deductive capabilities are here considered to be irrelevant as such. They may be used during the process, f.ex. to dynamically create new representations, but the focus is still on the conceptual representations.

relations. From the point of view of this framework the concept "knowledge representations" is reserved for the KR level whereas the knowledge representation techniques used at other levels are ignored, e.g. the knowledge representation techniques used during NLP or for the mappings.

3.3.3 Text Representations

The text representation level contains representations extracted from the texts with formal (syntactic) methods. These include content representatives extracted from natural language text using NLP and statistical methods as well as document structures and formal properties extracted utilising mark-up information. Text representations are used to represent texts when large amounts of texts must be searched through efficiently or the search process is based on some derived properties of the texts -- only the text representatives are compared to the request, which minimises the effort needed. If some text retrieval system do not process texts before retrieval but use text skimming methods [86], the text representatives exist only implicitly. In these cases the concept "text representatives" includes those features that can be searched (extracted) using the skimming methods.

Structural properties tagged to texts are fairly easy to extract. Regions tagged as author field, date, or title of a document are easy to map against concept classes of a knowledge representation system and their contents can be defined as concepts of these classes with a high probability. Also lists, tables, and rules tagged in text can be utilised efficiently for several purposes. In some domains the heading hierarchies of documents or books are as such descriptions of domain taxonomies. In all these cases the marked segments of texts are fairly easy to map against some model of the domain, but the contents of the regions may require NLP before it can be unified with individual concepts of the knowledge representation level.

Content representatives extracted from text with statistical or NLP methods vary a lot according to the level of processing used. In the beginning of the NLP scale there are strings limited by space characters, i.e., technique of traditional inverted files with use of all words of a text as its' content representatives. The next levels include truncated words and word baseforms, compound words, and phrases as explained earlier. In the other end of the scale there are normalised knowledge representations extracted from the text using morphological, syntactical, and semantical processing. When also pragmatic issues about the context are taken care of the content representatives have reached the level of knowledge representations. Typically the text representatives extracted automatically from text do not reach this level, because of their string dependency problems described earlier (such as synonymy, polysemy, and ambiguities). For example, terms extracted from text cannot be used as such, as index terms of the knowledge representation level in the traditional meaning of index terms. Thus some kind of mapping between text representatives and knowledge representations is needed.

3.3.4 Mappings

The model contains three levels of mappings, of which only the mappings between the knowledge and text representations are interesting from the point of view of this work. These mappings are used for mapping the knowledge and text representations (or the texts represented by them), i.e., to overcome the gap that is left after extraction of text representations. The mappings can be implemented in various forms, such as lists of classifications for each text or a thesaurus linking words used as text representatives to index terms used as knowledge representations[14]. Thus the mappings may exist as pre-defined data storages before any texts are stored to a system (as a thesaurus) or they may be created during the processing when texts are stored to the system (as the classifications of a text). The mappings are then utilised during the text retrieval explicitly to map concepts to text representatives or implicitly by mapping text representatives to others (like in synonym expansion). If semantic NLP is used for extraction of knowledge structures from text, the mapping reduces to the matching of knowledge structures in a request to those extracted from texts. Document structures that are explicitly marked to the text have often clear pre-defined semantics and can thus be mapped easily to the knowledge representation.

The TRSs typically represent entities of all except the two topmost layers. The user ideas and mappings between them and their representations at the knowledge representation level cannot be represented. In addition, the interaction of a user and a text retrieval system may take place at other levels than the KR level. Thus the topmost mapping is excluded from the framework.

The two layers in the bottom, i.e., storage of texts as such and the mappings between texts and representations extracted from them, are out of the focus of this work. The mapping of a texts and text representations extracted from them is always explicitly available as a many-to-many mapping from texts to content representatives[15] or as an one-to-one mapping from text segments to their contents i.e., texts. The data structures and techniques for optimising the efficiency of these mappings are uninteresting from the point of view of this framework -- it is sufficient to know that they exist and can be accessed -- and thus excluded from the framework.

3.4 A Framework for Representations

The various strategies for processing and retrieval of texts can be characterised by the methods applied on the intermediate layers, the levels of representation highlighted in the Figure 1. These levels are somewhat independent in the sense that replacing a method with another at one level does not necessarily force the change of the methods at another level --

---

[14] Formulated also as ambiguous relations between lexical and semantic units, between lexems and sememes, or between symbols and messages (see also [193]).

[15] Such as lists of keywords extracted from each text or as an inverted file containing terms and for each of them, references to their occurrences in the texts.

although, in practice, they typically are used as certain combinations. These three levels and the idea and text layers are later on referred to as the framework for text retrieval systems. In later chapters, the term "mappings" refers by default to the mappings between the knowledge and text representations.

Figure 2 represents the framework in a descriptive way. Ideas of the authors/librarians (on the left) and possibly fuzzy ideas of the retrieval users (on the right). Knowledge representations containing possibly a separate definitional component describing the overall structure of the world/domain (on the right) and representation of the ideas and entities relevant for text retrieval (on the left). The mappings between knowledge representations and text representations are represented as a layer of its own. The text representations containing content representatives extracted from the texts (on the left) and document structures marked up explicitly (on the right). On the bottom there are the texts organised for retrieval purposes as units whose size is meaningful for the domain in question.

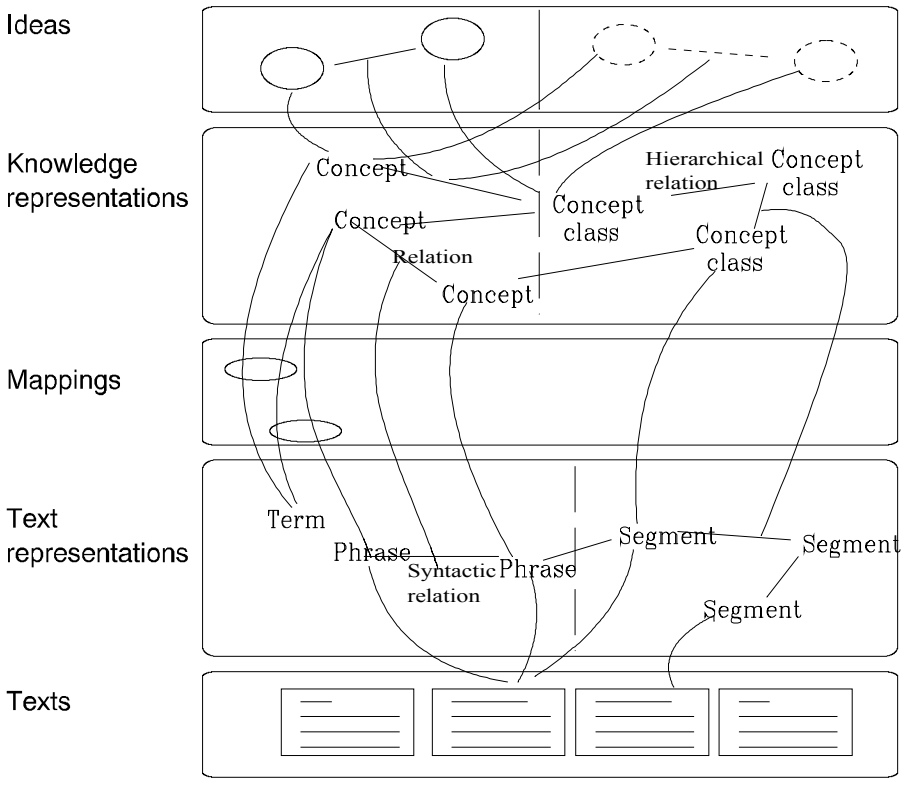

Figure 2. Descriptive figure of the contents of the TR levels.

Both the KR level models and the mappings are typically domain-dependent and organised for text retrieval purposes of a single domain, such as the medical subject headings (although there exists some emphasis on building large or general KR models applicable also for text retrieval purposes, such as CYC [12, 129, 202]). As the things to be represented at the KR level are domain-dependent, the mappings between them and the text representatives are also domain-dependent. The meanings of terms and phrases as well as contents of synonym lists or thesauri vary according to the domain. Instead, the methods

used for extracting representations describing texts are typically at the text representation level of the framework that utilises only formal (syntactic) methods and is thus domain independent. Even though the extraction of text representatives is language-dependent. If the different use of language in different domains is considered, also the text representation level methods are somewhat domain-dependent. At least domain-dependent terminology has to be included although the use of same terms in different meanings is represented by mapping-level representations and is domain-dependent, as stated above.

3.5 Other Dimensions of Text retrieval

In our framework the focus is on the syntactic-semantic scale of the representations; the methods applied for text representations extracted from texts by syntactic methods is much different from the semantic representations that aim at easy mapping to mental images of the users with string-independent (natural language independent) representations in contexts. From our point of view other relevant dimensions in analysis of text retrieval and its representations include

- the function of the representations,
- the interdependence of the representations, and
- the nature of the interaction (searching vs. navigational browsing).

In text retrieval systems different representations are used mainly for representing the content and structure of the texts, for representing a model of the world or a domain, for representing the users information need as a request, and for representing the users. Of these, the representations for content and structure of the texts use mainly the text representation level of the framework, although some NLP-based systems produce content representatives that clearly are representations of the KR level. Analogously, most of the world or domain models can be classified as KR-level representations. User modelling is out of the scope of this work, but the representations used for it are mainly those of KR level. The representations used for user request are closely related to the nature of the interaction discussed after a while.

The degree of interdependence of the representations is closely related to the other dimensions. From the point of view of the functions, the representations used for world and domain models at the KR level are highly interconnected both the definitional components containing general taxonomies and the data representations containing interconnected knowledge representations. Text content representatives at the text representation level are typically fairly independent although the content representations reaching the KR level contain some internal structure. Most of the structural properties of the text representation level are used as interdependent structures during the text retrieval whereas some marked fields are used only to pick up segments of texts without utilising their interrelations. Thus majority of the representations at the text-representation level are independent whereas almost all of the KR-level representations are interdependent.

The interaction methods are highly dependent on the interdependency of the representations. Independent representations can be searched efficiently, but cannot be navigated. The identity of highly interdependent representations depend (partially) on their interrelations, which makes querying hard, although not impossible. At the KR level the interdependent representations aim at string independence making the querying even harder. Even though, if the content representations have reached the KR level, they can be used for searching similar representations, although the unification requires matching contexts and estimation of the similarity that has been found difficult in the machine learning and automated classification research. During navigation or browsing the users have more information about their requests, the contexts, and their preferences than the system has based on a single query. Thus the users are supposed to make better decisions about the relevance of closely related concepts to their mental images of the information need than the system. Thus the most suitable interaction method for the KR-level representations is navigational browsing and the searching paradigm is the major interaction method for text representation level representations. The semantics of document structures at the text representation level are not so dependent on their interconnections, and thus the browsing paradigm describes best their nature during text retrieval. The interaction in hypertext systems is analysed in detail later on. In general, all representations can be used for mediated browsing although browsing independent content representations in an alphabetical order is less attractive than browsing interdependent knowledge representations. Based on these outlines the relation of user requests and the interaction methods is clear; explicit representations for user requests are used with the search paradigm (i.e., a query), but not with the navigation or browsing paradigm. Thus the use of navigational browsing is dominant at the KR level and the searching paradigm is most common at the text representation level.

With respect to the dimensions of the information exploration [237] (see page 25), the structural responsibility is closely related with what we call the interdependency of the representations -- the user is concerned with the structure of the interdependent (i.e., KR level) representations but not with the independent text representations. Target orientation is close to what was above called as the interaction method; searching at the text representation level and navigational browsing at the KR level are typically used for finding a definite target whereas mediated browsing (and mediated searching) do not have a definite target. The dimension called interaction method, in [237] separates between descriptive and referential paradigms that are also closely related with the interdependency -- descriptive interfaces are natural for independent representations whereas referential interfaces are natural for interdependent representations, especially if they are string independent, as the KR-level representations aim at.

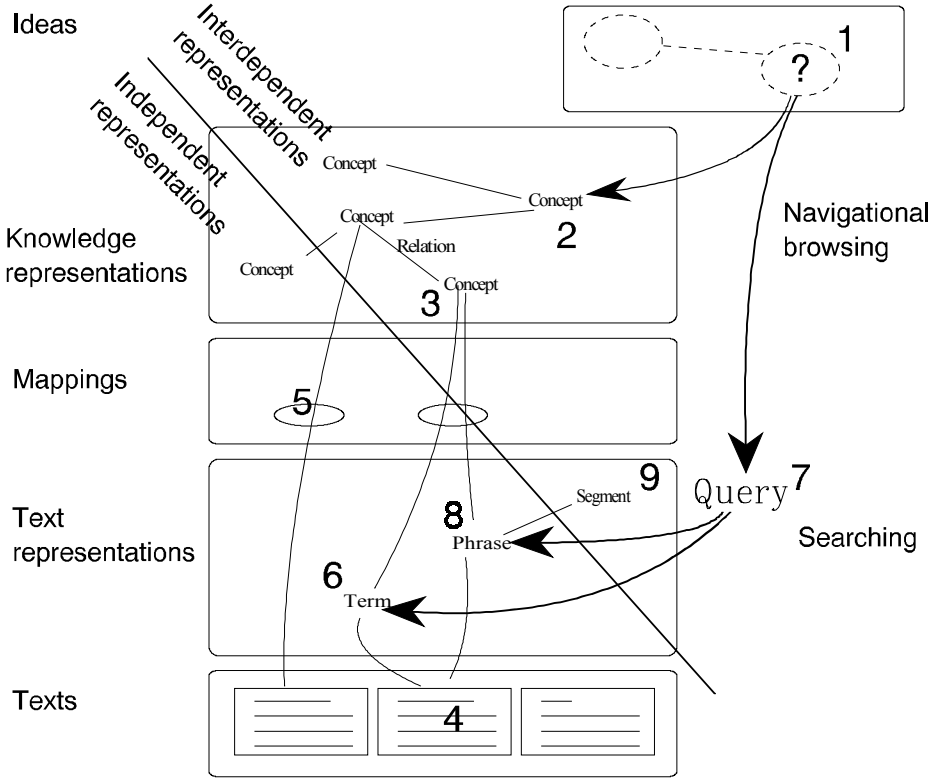

Figure 3. Searching and navigational browsing.

Figure 3 represents the use of searching and navigational browsing in text retrieval. The usefulness of these paradigms is defined roughly by the line dividing the representations to interdependent and independent representations. The same line represents also the rough division of the representations to those used for world or domain models and those used for representing content and contents of the documentation. In the beginning of the text retrieval process the users have information needs (1) in their minds. By navigating the knowledge representations on the KR level (2-3) the required classifications, topics, or concepts are found (3). The texts describing these concepts (4) can be linked directly (5) to the concepts as texts classified under subject headings or the system may use some internal mappings to text representatives (6) which are used to represent the concepts and are present in the required texts (4). The other main alternative is to describe the information needs in terms of text representatives, by defining queries (7) that are matched against content representatives (6, 8) in required parts of texts (9). The texts needed (4) are found as the texts containing representatives satisfying the queries. The different text retrieval approaches are analysed further in the following chapters.

4 ANALYSIS AND OVERVIEW OF TEXT RETRIEVAL APPROACHES

This chapter represents the different text retrieval approaches relevant for this work and analyses them from the point of view of the framework presented in the previous chapter. This first sub-chapter describes typical features of conventional approaches including information science methods and the numerically oriented IR methods, such as vector space and statistical methods.

4.1 Conventional Approaches

The use of knowledge and text representations in conventional text retrieval systems are represented in Figure 4. In the manual book/article library paradigm the text processing is performed by a librarians (1) that create the mappings (2) from texts (3, books, articles, or reference cards) to the classifications or index terms (4) of the KR level model used. Formal properties of the texts, such as name of the author, date, language, etc., are also extracted to the text representation level or full-text indexing may be used (5).

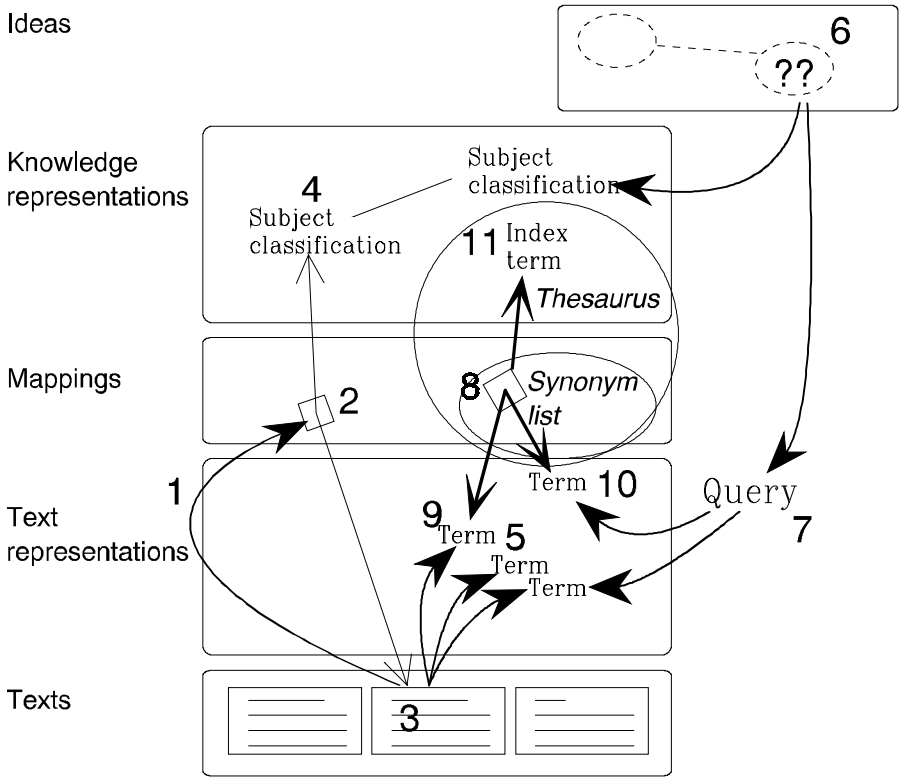

Figure 4. Conventional approaches to text retrieval.

In the text retrieval phase the users try to find required information (6) by browsing the KR level, the classifications or index terms (4), and listing texts classified under them (using 2 to 3). The users may also use full-text search capabilities at the text representation level, i.e., to search texts satisfying a query (7). The results are represented at the text level as sets

of texts and set operations, such as union and intersection (i.e., Boolean OR and AND), can be used for combining the results.

The searches may also be extended using synonym lists (8), a kind of limited thesaurus at the mapping level. This means that the mappings from KR level to text level are used for listing the appropriate text representatives (9, 10) without explicitly pointing out the concept (11) defined by the representatives. Many thesaurus systems define the concept as the preferred term from a set of synonyms. A thesaurus lists often also the senses of a word and alternative words used as synonyms with respect to the sense groups, i.e., concepts. Thus it describes the many-to-many mapping between the terms at the text representation level and the concepts (word senses) at the knowledge representation level.

Some (e.g. medical) domains have very well defined structures that are embodied in a thesaurus. A sophisticated thesaurus may contain relations, such as *broader term* or *subconcept,* as well as several domain-dependent relation types.[16] This kind of thesaurus structure can be seen as a combination of a classification hierarchy, a set of non-hierarchical relations between the concepts, and mappings to terms representing the concepts. It contains elements both from the KR level and from the mapping level. A thesaurus like this is often accessed by giving a word or a phrase to a thesaurus system. Later on the references between concepts may be browsed until the required concepts are found and a query will be created using the terms of a concept. This kind of thesaurus [54, 69, 111, 115, 119, 148, 175, 177, 223, 234] is close to a semantic network and provides sometimes users with menus [104] or graphical browsing environments [54, 115, 148] to explore the structure of the domain by browsing the thesaurus. The retrieval paradigm may also be based on the similarity of a user query and text representations based on graph similarity or distance of the concepts [111, 148].

Some IR approaches have utilised knowledge representation languages for implementing concept networks or frame indexing systems [90, 148, 240, 248] . The index frames with pre-defined facets are used similarly to faceted classification schemes to represent the features of a text as literal facet values of a frame. The indexers may also assign a set of (possibly interlinked) frames to describe the text. These frames are chosen from a set or a class hierarchy of domain-specific frame structures. The concept indexing systems are analogous except that the frames are replaced with objects or literal concepts with no internal structure. Some of these approaches support graphical navigation of the representations [148].

The difference between use of (manually) assigned index terms and full-text searching is clear based on the separation of the knowledge representation and text representation levels; The full-text paradigm uses the words of the text (almost) as such as the content

[16] There exists also several standards for thesauri [124, pp. 29-33], such as ANSI (ANSI Z39.19) and ISO [76] standards. A thesaurus, considered here as a sophisticated one, contains a variety of domain dependent relations between concepts, not only fixed relations between terms.

identifiers whereas the index terms assigned to a text may be present in the text in some modified form or may even not exist in the text at all. The traditional meaning of concept "index term" associates it with a semantic content so that a text may be indexed using an index term even if the term does not occur in the text. Thus it cannot be a text content representative extracted from the text with no semantic and pragmatic information. The assignment of the index terms requires some pre-defined mapping and the index terms can be described as knowledge representations of a domain. Anyhow, the nature of index terms is not quite that simple. They are typically defined as concepts represented by multi-word terms. The most common way to map these concepts to content representatives is to use the single word terms of the index term as the search request, i.e., to search for texts where all the words in an index term occur. Thus the way to use index terms is more related to the mappings than the ways to use subject classifications, for example, and the mixed role of index terms can cause ignorance to some aspects of their use, such as their organisation into browsable structures.

IR research has improved the efficiency of full-text retrieval systems by several means during its short history. The major concern has been matching user requests with the text representations where as the knowledge representation and transfer have been more or less ignored. The improvements have focused on numerical methods used for matching the vectors representing user queries and texts, and lately, on query reformulation based on relevance feedback [71]. The early improvements include research on different search tactics [14], proximity search for related words [186], weighting of terms in queries and texts according to their frequencies or relevances and ranking of results correspondingly [104], quorum-level search [33], matching term vectors representing queries and texts based on various vector operations [186, 235], matching based on fuzzy sets, and use of synonym specifications and term truncation on query expansion [186]. Vector space models have been popular lately including use of term relevance, i.e., probabilistic retrieval models, as well as context vector representations [67] and other variations [208]. The Extended Boolean model [245] generalises the Boolean retrieval model and the vector space model. The ideas of vector space models have also been used for comparison content/contents of text excerpts [188].

Vector space models operate explicitly in the text representation level (especially with the text content representatives) as the conventional Boolean-based querying. The major advantages of these approaches are a (partial) domain independence[17] and ability to use fairly fast methods for extraction of the content representatives during text processing (i.e.,

[17] Statistical methods are typically justified for a domain based on a large text corpus containing representative text material of the domain. Similar corpus based methods can be used for document clustering, thesaurus construction, and extraction of indicator terms. If the coverage of the corpus is inadequate, the results may be misleading, like "term *Jesuit* is the best indicator that a news story is about a murder", as in the corpus used in MUC-3 [128] tests.

database input). The problems occur during the retrieval process when the user request should be mapped against the low-level text representatives, words or word stems, via the query language. Lots of work is spent on this query reformulation process, to create the mappings between a user request and the text representatives corresponding to it. This mapping can be created based on feed-back from an experienced user and the texts can be located. The major advantage of this process is the flexibility; in case of full-text search there is no predefined set of concepts to be searched for, as in case of automated indexing or controlled vocabulary. Unfortunately the mapping created during text retrieval process is typically lost after the retrieval session, the creation of the mapping takes always some time, and may even be impossible for casual users. If the term weight vector or other parameter values produced by a feed-back process are stored as a macro defining a concept, there still exists the problem to communicate the semantic contents of this concept to other users. This would require a definition of the concept in the form of relations to other concepts, not only as a mapping to terms with no context on the knowledge representation level. Thus no support for knowledge transfer to users or support for browsing and navigation is provided.

Use of NLP and AI techniques have been studied widely for automation of the conventional librarian tasks [61, 88, 172, 202]. Automatic classification has been studied based on the vector space model and clustering methods [186, 236] and based on simulation of human workers [30]. Computer aided and automated indexing has been used for choosing the best single terms for indexes based on term frequencies, probabilistic term weighting, and relevance feedback [186], as well as formation of index phrases based on use of template phrases, statistical, and linguistic methods [51] as also in SIMPR. Automated abstraction can be based on term frequencies, indicator phrases [19, 62], and anaphoras [21]. Automated thesaurus construction has been experimented based on co-occurrences of terms [104, 112] and based on user search behaviour [77] (see also [9]).

Most of the commercial text retrieval systems on the market are still Boolean-based. The statistical methods have not reached any major market share, whereas some products with concept definition methods are fairly popular, such as ITMS, Personal Librarian, Status/IQ, and Topic [219].

4.2 Natural Language Processing Approaches

4.2.1 NLP-Based Text retrieval

The use of natural language processing for information retrieval purposes is obvious; many of the problems in contemporary TR systems are due to inherent features of language. Thus the incorporation of linguistic concepts is essential in order to achieve intelligent text retrieval [48]. The strategies adopted vary a lot according to the level of processing adopted and according to the phase of processing where it is utilised. The most common way to use NLP is to use it in text processing and thus equip users with better tools to retrieve the

information. The retrieval process may then either utilise NLP or use the conventional methods. NLP can also be used for closely related purposes such as classification of information based on the content, for example, in classification, filtering, and rerouting of messages [29, 101, 128]. Figure 5 represents different strategies of NLP-based approaches from the point of view of our framework for representations.

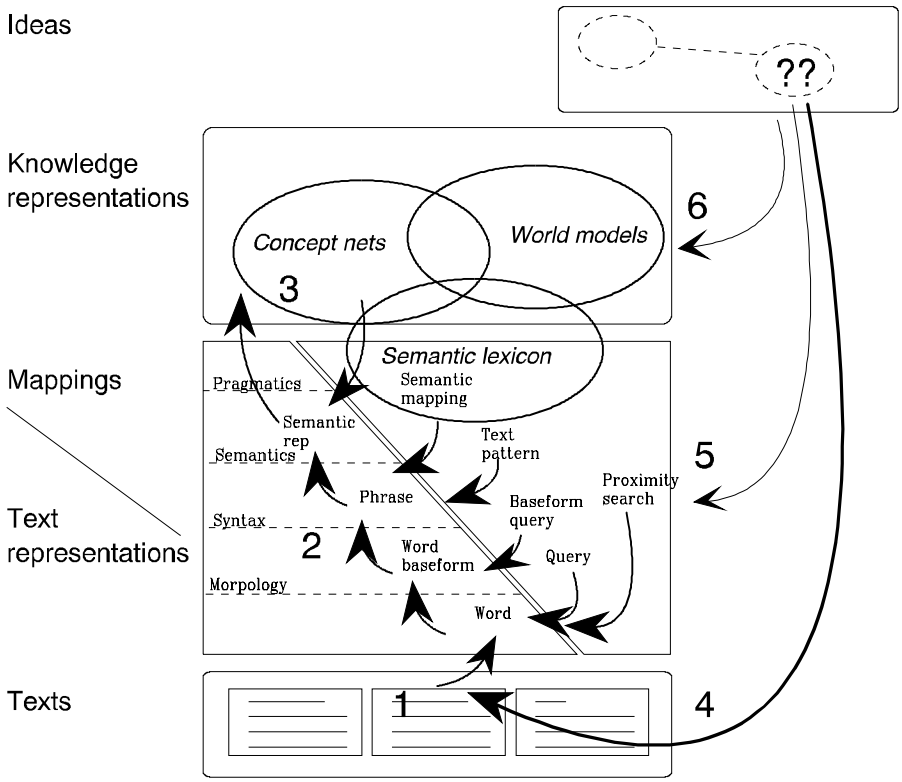

Figure 5. NLP-based approaches to text retrieval.

Typically the processing phase is enhanced by NLP methods in order to make the retrieval easier. The processing follows the levels of NLP and is represented as a path of arrows (2) from the texts (1) via text representations upwards to the knowledge representations (3). With no NLP at all we are able to use statistical methods on word forms. With morphological (lexical) analysis and lexical disambiguation the systems can use word baseforms or compound words [119] as text content representatives. In an inflective language, such as Finnish, the use of word baseforms as index terms instead of all word forms decrease the number if index terms (about 50% in Finnish [160, 161]) and gives relief to many of term truncation and wild-card problems during retrieval thus improving precision [4]. Also multi-part compound nouns can be permuted during indexing.[18] This enables users to retrieve texts containing noun "tekstinhakujärjestelmä" (Finnish, tr. "text retrieval system") when the request contains one of the terms "teksti", "hakea", "järjestelmä", "tekstinhaku", or "hakujärjestelmä". ("text", "to retrieve", "system", "text

[18] An idiosyncratic feature of Finnish is that words combine to form new concepts. In running newspaper text more than 40% of word baseforms have been reported to be multi-part compounds [160].

retrieval", and "retrieval system" respectively) improving recall [4]. Lexical categories can be used instead of stop word lists to reduce the number of indexed terms. After syntactic analysis the syntactic features (head/modifier) of word baseforms could also be used for index term weighting although this information seems not to be suitable for text retrieval purposes as such.[19] The parse trees created by syntactic analysis can be used for index phrase extraction and thus for creation of structure from the non-structured stream of words [192]. Up to the syntactic level the NLP methods used are fairly domain independent and do not require domain-dependent knowledge.[20]

The NLP-based index term extraction in SIMPR operates on the morphological and syntactic levels. At first the texts are preprocessed and analysed morphologically. At the end of morphological analysis some 50% of the words are ambiguous but after applying constraint grammar rules in the lexical disambiguation phase 94-98% of word forms have a unique interpretation (with negligible error rate, less than 0,3%!). The same constraint grammar formalism is also applied to syntactic analysis [107]. A rule-based system called MIDAS is then able to extract and reformulate multi-word index terms[21] from the linguistically analysed texts [72]. The extracted index terms represent the original texts reasonably well although manual validation is still required. An alternative index term extractor uses tree structured index terms in order to maintain syntactic ambiguities within index terms [201].

After the syntactic level the NLP processing requires pre-defined mappings to reach the semantic and pragmatic levels as well as some prior organisation at the KR level [85, 168, 248]. The mappings from text representatives to the knowledge representations may be pre-defined in the form of semantic lexicons that are used for mapping terms or syntactic structures into meaning representations. The representations used at the KR level contain typically some kinds of hierarchical world model for describing the general structure, such as semantic categories or classes or conceptual hierarchies or categories [29, 44, 74, 79, 85, 100, 101, 167, 168, 170, 180, 248]. The text processing may go through intermediate logical forms as well as to utilise statistically oriented methods, such as calculation of evidence based on related concepts [29, 241]. As the result of the processing, a text is assigned a set or a network of KR-level representations as its content representatives (3). The representations used for this purpose include a set or a network of interlinked concepts [29,

---

[19] An internal SIMPR report by Alain Smeaton, Dublin City University. Also the modifications to NLP grammars in project CLARIT emphasize that the heads of clauses are not necessarily the most important parts of the text [57].

[20] Except for the terminology (recall the chapter discussing natural language processing, page 23). Contemporary computational linguistics discard the disadvantages of using domain independent syntactic parsing for all domains. Anyhow, the syntactic methods are or contain at least language dependent parts.

[21] Examples of normalized index terms extracted from a car manual: "brake servo vacuum pipe", "fuel line disconnection", and "engine removal". The index terms include both noun phrases and other constructions according to the selected strategy.

100, 126, 241], (Sowa's) conceptual graphs [168], Schank's semantic primitives [23], as well as frame representations [79, 86, 128, 128, 130, 173, 222, 240, 248] and logical representations. Some of the frame representations contain pre-defined, domain-dependent attributes/slots analogous to those used by several non-NLP thesauri and index frame approaches as well as predefined concept classes, relation classes, or categories in some NLP systems [128, 168].

The knowledge representations extracted from text may be linked to the main knowledge base [29, 85, 100, 101, 180] or may be stored separately. This process may be utilised for rough knowledge acquisition from text [74, 134, 144, 162]. If the process were complete, it would be close to a reversible process and the processed texts could be reproduced close to their original form.[22] This can be utilised, for example, in automatic abstracting and in question answering systems with natural language response [79, 101, 168, 180]. Natural language generation systems [7] perform only this reverse process (without forgetting planning, discourse structures and other high-level activities).

NLP processing is typically utilised also during text retrieval. The requests (4) are processed similarly to the processing of texts and the results are matched at a common level of representation. The process follows the traditional information retrieval strategies: Texts are processed and text representatives, i.e., internal representations of the text contents, are stored to system database. User request (4) represented as a natural language statement or a question that is processed similarly and is matched against the internal representations derived from texts, at a feasible level of NLP. At the morphological level the processing of a user query means returning the words into their baseforms and transforming to the query syntax, i.e., a NL interface to an traditional IR system [8]. At the syntactic level matching request phrases against phrases extracted from texts [9, 42, 195, 199] tries to avoid false matches of co-occurring terms in different phrases. Matching semantic representations in form of matching logical representations, frames, or graphs [10, 168, 178] is supposed to help to cope with synonymy and polysemy problems occurring in matching terms instead of meanings. Even though this matching has problems e.g., normalisation, transformation, and unification of the representations. Some of the systems utilise relations between concepts to define texts relevant for related concepts or a query [10, 29]. This is based on relevancy propagation rules or numerical calculations and weighted relations. The weighted relations are used in retrieval time calculations where the related concepts are used as "sub-query macros", whereas the propagation rules are typically used during text processing phase for creation of pre-defined links between texts and concepts. With respect to the browsing and navigating facilities the systems have very different functionalities. Although the results of NLP processing are typically used by matching generated representations, some of the representations can be used for navigation and browsing [12, 123].

---

[22] According to Doshita et al. [162] the main problems in automatic knowledge acquisition are more in knowledge representation and inference than in parsing.

NLP text representatives can be utilised also without NLP during retrieval. At the text representation level the texts containing a given term in any inflected form can be found using text baseform indexes produced with NLP as well as user-given phrases can be matched against phrases extracted from texts (5). At the KR level the structures extracted from texts can be browsed or searched (6) and texts where they were extracted are retrieved. The view taken in this framework focuses on the human interaction with the model as the primary method for matching representations extracted from the texts with the users' information needs. This avoids the normalisation problems of knowledge representation. It relies on human competence to recognise the requested concepts that requires representations to efficiently support the navigation.

NLP can also be used only during retrieval. This means that the retrieval system processes the texts during retrieval process and matches the request with the generated representations. In order to do this within a limited time, the system must be able to select the (pieces of) texts to be processed. This can be implemented by 1) transferring the request to lower level text representatives that are used for 2) searching all texts that may match the request, and by 3) filtering the false matches at each level of processing until 4) the set of representatives can be compared to the request at the chosen level. If the phases 3-4 are ignored, the resulting strategy can be used, for example, for searching texts containing any of the inflected forms generated from a request term. This is useful if the truncation of a word cannot be defined easily or produces extra surface forms (of other terms) as well as in languages, such as Finnish, where the number of inflections is high.

4.2.2 Problems with NLTR

At first the use of NLP and NLU (natural language understanding [194, pp. 660-667]) seems like an ideal method for text processing and retrieval. It tries to manage the problems due to the inherent features of natural language, encode the full content of the text to knowledge representations, and is able to response any natural language query with texts which are hopefully relevant to it. But there are still lots of problems to overcome.

Natural language processing is a knowledge intensive task on any level of processing. The knowledge needed may include knowledge about the morphological, syntactic, and semantic properties of tens of thousands of words as well as knowledge about pragmatics, discourse, and context, and some kinds of common sense knowledge. With respect to the levels of representation NLP uses knowledge representations in text representation, knowledge representation, and mapping levels. The representations used for morphological and syntactic knowledge (at the text representation level) are specialised for efficient manipulation of large amounts of language. Typically lots of the knowledge is organised around words in a lexicon and around classes of words that can be manipulated similarly, e.g., manipulated by the same set of rules or automata. Knowledge may also be organised as hierarchies that determine the automata, syntactical, or especially the semantical category or

a word or position in a taxonomy. Explicit hierarchical representations are often used as a framework around which the large knowledge bases are organised. They are useful both because they classify and organise the knowledge and because they provide default values and possibilities to represent exceptions and thus capabilities to smoothly add the amount of knowledge [129]. Also statistical methods have been used for describing syntactic knowledge.

At the text representation level the richness of natural language leads to problems in acquiring and updating the huge amount of knowledge. The major problems on morphological level are unknown words [78, 180] and terminology as well as domain-specific use of language [100]. Due to the flexible nature of the language even large lexicons contain only a portion of all words and word senses used in a domain. Especially new names and abbreviations are taken to use all the time [78, 108]. At the syntactic level compound nouns and other phrases, ellipsis, and scoping [141], as well as anaphoras [21] make the parsing hard. In a comparison with the later levels, the syntax analysis is more portable than semantic and pragmatic analysis. At all levels, the incomplete knowledge and inherent features of natural language lead to lexical, syntactic, and semantic ambiguities [196]. In addition, NLP requires lots of processing time and resources due to the computational complexity of natural language [45, 191].

The major problems with NLP are encountered in the semantic and pragmatic processing. First, although the manipulation of single sentences has been successful to a large extent, the problems with discourse and story understanding need still lots of further work in solving anaphors and other references to previous parts of a story and to the world knowledge. The second major problem is the representation of the world knowledge needed for semantics and pragmatics; the knowledge base is huge even within limited domains [186], but extreme in unlimited discourse area [202], and also some kinds of common sense reasoning may be needed [44, 129, 143]. Some projects, such as CYC [12, 129], have been working on automatic knowledge acquisition from texts for years but have not reached sufficient level of world knowledge so far. The third major problem is the ambiguity and domain-dependency of the mapping from terms and phrases to the knowledge representations [117], where known words can map to different meanings in different domains and also within the same text. The problems of translating texts to a uniform "canonical" semantic representation[23] is analogous to the problem of creating a universal language, a semantic interlingua[24] as was attempted in many machine translation projects earlier.[25] This is related to the problems of disambiguating and matching semantic

[23] Typically via some logical form to a frame structure. See for example [66].
[24] Often in form of a thesaurus [104].
[25] E.g. EUROTRA [114] that at first attempted to use one semantic interlingua but later on moved over to bilingual translations between the separate European languages.

representations created for texts, problems in defining the matching criteria,[26] and to the cost vs. accuracy trade-off of the matching operation. If the searching within the knowledge representations is based on string matching, the string-dependency problems are still present in the system [186]. Successful matching of user requests with the stored representations depends also on a users ability to name the requested information so accurately that the request can be transformed to the same representations at KR level.[27] Also the management problems of large KR models and text representations are massive; the storage space required for internal representation of megabytes of text is also likely to be megabytes requiring specialised storage and indexing as well as expensive operations when matches against user request are performed. Finally, most of the current systems for deep analysis are characterised by inefficiency of processing and are typically applicable only to analysis of amounts like 20 sentences to 20 pages of text per hour.

Whereas the purpose of NLU systems is to produce full representation of the text the text retrieval oriented NLU or NLP systems aim at extracting only the key concepts of the texts. These systems utilise some semantic processing features of NLU systems. They operate typically in a limited domain that enables them to resolve some of the ambiguities and reduces the amount of required knowledge. This approach gives some relief to the problems encountered when pure NLU systems are applied to text retrieval purposes. An alternative approach is not to use semantic processing, but to utilise syntactic processing in order to handle the structural features of natural language in extraction of index terms.

Due to the poor availability and efficiency problems of the contemporary NLP systems many TR systems have developed some shortcuts to tackle the problems of NLP (see the mappings, arrows downwards, in Figure 5) or alternatively improve the efficiency by skimming or otherwise processing only important parts of the texts, e.g. the headings or pre-selected text segments [10, 29, 60, 86, 101, 130, 180, 221]. In English the morphological analysis is often substituted by a stemming or truncation algorithm. The ability to recognise phrases and compound words is typically taken care of by proximity search. Syntactic structures and concepts are often recognised by templates and pattern matching algorithms. The commercial text retrieval systems typically utilise this kind of rough approaches to manage large text bases [219]. Also the conceptual indexing systems typically map conceptual hierarchies directly to templates of word patterns. This means that a domain expert - a person having vast knowledge of the domain and it's concepts - defines the

---

[26] The contemporary research on automated classification and machine learning consider the selection of relevant features to be used as the classification or unification criteria to be one of the fundamental problems. The connectionistic approach gives an alternative view to this issue [152].

[27] Mainly due to these two reasons -- need to name the requested information with sufficient accuracy as well as problems and context dependent criteria of matching the semantic representations -- the use of knowledge representations as a semantic interlingua for matching representations created from users' requests and texts is considered to be secondary to users direct access to the representations.

mappings between concepts of KR-level text representatives at the lowest level of text representatives.

4.3 Structure-Oriented Approaches

4.3.1 Mark-up and Document Structures

In the previous section the focus was on the utilisation of natural language processing techniques for TR purposes. There the term "structure" referred to the syntactic structures of sentences, phrases, etc. In this chapter the focus is on explicitly marked, i.e., tagged, structures of documents and documentation such as the separation of a documentation to individually manipulated documents and the partition of a document to headings, paragraphs, tables, etc. These kinds of explicitly marked structures are also utilised for information retrieval purposes as well as to aid in language analysis. Simple examples include utilising the structure of reference records in bibliographic databases and creation of back-of-the-book index for text processing documents by collecting the phrases marked up as index entries. This information can be used for phrase indexing of text databases as well.

A vast amount of different mark-up languages has been developed for text management, processing, and formatting purposes. The early mark-up conventions in text formatting systems contained implementation-specific commands for the printer to move the printer head, to change font, to use underline, etc. Later on the mark-up languages have abstracted the markers to a higher level, to describe the starting point of a heading, of a footnote, or an emphasised part of text. One of the most important standards in this area is Standard General Mark-up Language (SGML, ISO 8879, CALS)[28] for document interchange format. The objective of SGML is interchange of documents between separate document processing systems (text editors, publishing systems) and other systems utilising the same documents (information retrieval systems, translation workstations, management information systems, etc.). SGML is system-independent, device-independent, language-independent, and application-independent. In fact, SGML is a meta language for mark-up languages. It describes the syntax of the mark-up languages used in information interchange. Thus it does not set limits to the information that can be text as well as figures, bitmaps, audio, or video images [103, 158].

In many domains most of the documents have a uniform hierarchical structure or a limited set of uniform structures. In office environments the letters contain some header information including names of the sender and receiver, date, and title. The header is followed by text containing headings and paragraphs and possibly a footer containing signature and other additional information. Similarly the orders, invoices memos, and other

[28] ISO = International Standardization Organization. CALS = the Computer-aided Acquisition and Logistic Support Initiative of the U.S. DoD, Department of Defense. See also [55, 103] and British standard (BS 6868).

documents conform a set of structural standards. In structured document database systems [110, 138] the system supports writing, editing, and querying structured documents.

The emphasis on document modelling has several goals. There exists architectural models describing structural properties of the documents, models focusing on facilities for syntax-directed editing [56], and approaches that consider document modelling as a special case of data modelling [184]. The idea of defining types and operations for parts of documents [184] is in some respect parallel with the ODA/ODIF document architecture standard, Open Document Architecture and Interchange Format (CCITT T.411, T.412, T.414-418, ISO 8613). The objective of ODA is to standardise the representation and interchange of documents between different systems using different internal formats. ODA separates the structure and the contents of a document. In ODA both the logical and layout structures of a document are defined as tree structures. Documents satisfying the ODIF interchange format definition can be transmitted between systems ensuring similar appearance of the documents in heterogeneous environments.

SGML described above includes also a component for defining hierarchical document structures, namely document type definitions (DTD). DTD is used for defining the allowed sub-components of each tagged component analogously to the definitions in ODA. In comparison to ODA/ODIF, SGML is purely a syntactic tool with no definition of the procedural semantics for each DTD component that would be needed for drawing the documents in a similar way in each environment, for example.[29] In this respect SGML is not limited to a narrow set of interchange formats as the ODA application profiles in use,[30] but leaves much responsibility for the applications.

Figure 6 represents structure-oriented approaches for text retrieval from the point of view of the framework for representations presented in previous chapter. During retrieval the structural properties of texts are usually utilised by browsing knowledge representations although searching strings from specified fields is also very popular.

Structural properties of texts are fairly easy to extract. Text segments tagged as author field, date, or title of a document (1, 2) are easy to map against concept classes of a knowledge representation system (3, 4) and their contents (5, 6) can be placed as concepts of these classes (7, 8) with a high probability. It is also possible to create relations between concepts (9) based on the relations of tagged fields (10) in text structures, such as lists, tables, and rules. For example, rules for an expert system can be extracted from text directly based on tagging [147], i.e., the same document can be used for textual representation and a computable model of that domain. These linkages between concepts and texts are easily utilised in text retrieval applications. The concepts (and relations) recognised from marked segments can be used as KR-level text representatives similarly to the use of deep NLP

[29] Although this kind of information can be interchanged within SGML notation and standardization of this interchange has been initiated [102].
[30] For example, ODA Q/112 supports text, raster figures, geometrical graphics, and fonts.

processing results. Typically there exists an one-to-one mapping (11) from text segments to concept classes based on the pre-defined semantics of the fields. The problems are encountered in definition of the structures needed for a domain and especially on the unification of concepts to or searching concepts from a field contents. This requires some kind of NLP if the vocabulary and use of language are not highly restricted. General mark-up conventions, such as SGML, are very useful for this purpose [46, 66, 147]. They can be used even for automatic knowledge extraction from marked text structures [31, 147]. Similarly, document architectures can be used for document classification and retrieval [31, 46].

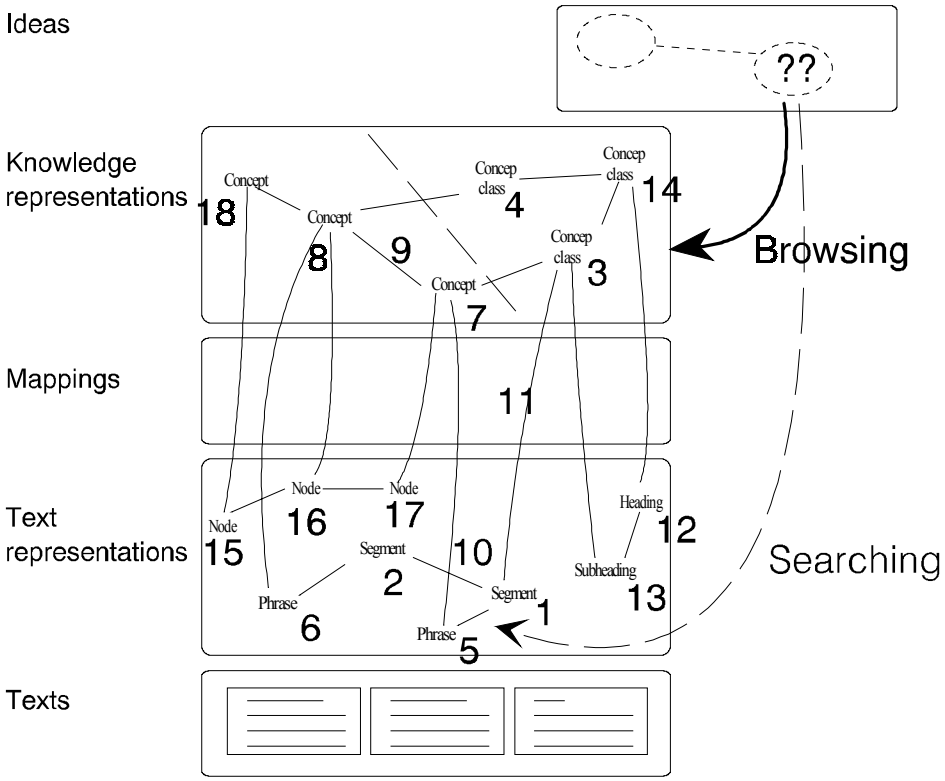

Figure 6. Structure-oriented approaches.

In technical domains the heading hierarchies[31] of documents or books (12, 13 in Figure 6) are often as such descriptions of domain taxonomies (14, 3), and appreciated as an important tool to access the information [174]. For example, technical documents may often contain main chapters, such as Introduction, Operation, and Reference or System Description. The next heading level under System Description may include Functional Description, Hardware Description, and Software Description, each of which contain sub-chapters and sub-sub-chapters like in Figure 7 (for further details see [92]). This kind of heading hierarchy structure is more oriented towards taxonomic description of the domain than towards sequential reading of the texts in a given order. The identity of text is easy to

[31] I.e., the tree structures defined by chapter-subchapter relations of a table of contents.

remember due to its' hierarchical nature. It is analogous to the conventional subject classification method; the texts describing a domain are organised according to one hierarchical view. Thus there exists an one-to-one mapping between the headings and a KR-level representation of the domain taxonomy that is easy to utilise in text retrieval. The problems with this kind of arrangement are similar to the problems of conventional subject classification systems, such as restriction to one view to the documentation, lack of cross-references, and effort needed for changing the structure or joining it with other taxonomies. Also the unification of headings strings with the concepts has similar problems as described in content interpretation of other tagged fields.

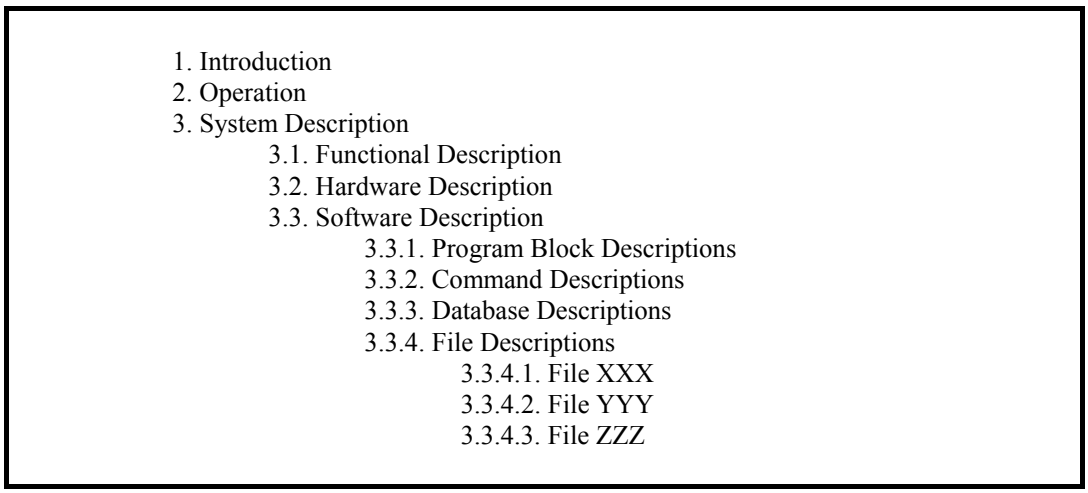

Figure 7. A heading hierarchy used as a domain taxonomy.

From text retrieval point of view the structured documents described earlier have features similar to the ones described above and can thus aid text retrieval. In document architecture approaches the analogy between the heading hierarchy and the classification hierarchy is formalised as procedural semantics of document segment manipulation as well as semantic expectations of named text structures and locations of particular information within a document. For example, journal readers form mental representations of a document's structure and organise isolated pieces of text into a more meaningful whole facilitating extraction of required information by non-serial reading [49]. In all these structure-oriented approaches the structural features of texts are rather easily mapped against some group or class of concepts at the knowledge representation level, but the contents of the regions may require NLP before they can be unified with individual concepts.

4.3.2 Hypertext

From the point of view of this work, the relation of hypertext and text retrieval is two-fold [230]. First, hypertext can be analysed as a text retrieval method. Secondly, the problems in creation of hypertext networks are somewhat analogous to the problems in text retrieval;

which text segments should be associated with a given hypertext node vs. should be associated with the user request. Here the analysis is based on the framework of representations described earlier.

Hypertext networks are often stated to be analogous to semantic networks containing concepts and links between them [27, 41, 54, 165, 237].[32] This means that there exists an one-to-one mapping from hypertext nodes (15, 16, 17 in Figure 6) to concepts represented by their titles (18, 8, 4). Also the links between nodes can be directly mapped against relations between the corresponding concepts. This direct equivalence of knowledge representations and texts[33] enables users to browse the knowledge representations and simultaneously see the texts describing the concepts; the KR level and text representation level are merged together. From the user interface point of view the smoothness and negligible time needed to "change between the levels of representation" produces excellent user interaction that is easily accepted by the users.

The problems of hypertext described earlier (e.g. disorientation and elaborate creation) are easily identified as the drawbacks of the unification of different levels. Problems to browse large hypertexts are typically solved by separating the KR level, and the text representation level again; the node titles and links between nodes are displayed separately from the texts, in a browser window representing the KR-level concepts and their relations. Furthermore, the missing definitional component at KR level is emulated by adding empty nodes to represent taxonomies and other hierarchical structures of the domain [165, 237]. The separation of text representatives and texts is implemented by adding attributes to the nodes as formal properties and by using words of the node title or body as content representatives in full text search [22, 139]. With these modifications the hypertext paradigm corresponds better with the levels of representation in the framework and will perform more adequately.

From our point of view the one-to-one mapping between the concepts and texts[34] (i.e., titles of nodes and nodes) does not only make the connection between KR and text representation levels easy, but also causes problems. First, it restricts the number of texts describing a concept to exactly one, neither more nor less. Secondly, each of the texts contains exactly one title, i.e., it describes always one and only one concept. These mean that

- two texts describing a concept must be assigned under artificially differentiated concepts (i.e., different titles),
- concepts cannot exist without a text that describes them,
- a text may not describe more than one concept, and

[32] Hypertext is also used as an implementation technique for sophisticated thesaurus constructions that were discussed earlier.
[33] Via one-to-one mappings from concepts to text representations and further from text representations to texts.
[34] More precisely, one-to-one mapping between concepts and text representations representing (one-to-one) the texts.

- a (new) concept must be created for each text.

As an (crude) analogy to the conventional subject classification systems this would mean that classifications may not be defined before some text classified under a subject exists -- not even any generalisations about existing classifications. Furthermore, there may exist only one text classified under a subject heading and locating the interesting subject would give us only one text describing it. Thus each new text would create a new classification and the number of texts classified would equal the number of classifications.

Although the presumption, direct equivalence of hypertext networks and semantic networks is greatly simplified,[35] the reasoning above points out the need for more structured hypertext systems where the levels of representations are separated.[36] We need also a classification of hypertext systems. In general, the semantic network (at KR level) defined by the hypertext nodes reflects very well the contents of the documentation. The disadvantage is that it reflects only the contents of the documentation; the organisation and structure of the network is poor and the network may be ill balanced describing similar concepts with artificially differentiated contents and lacking important or general concepts that are not explicitly represented at the texts, or even more, are not represented as separate texts. Thus creation of a sophisticated hypertext system requires separation of the KR and text representation levels by extending the mapping used at the mappings level from one-to-one mappings also to other ones.

4.3.3 DMG -- Hypertext Generation

Automatic hypertext generation has been one of the issues in recent hypertext research. The work of the author described in [226, 227] solves some of the problems in context of diagnostic advisory systems. The advisory systems are delivered to the users in the form of hypertext applications, where each of the nodes describes the test to perform and the users follow pass / fail links according to the results of the tests. This hypertext form provides also additional information to the users, such as figures of the correct test results in an oscilloscope display. In this case the tedious task to create and maintain this kind of hypertext application is solved based on hypertext generation.

The work is based on the idea, that the information needed for the individual tests is organised in the framework of hierarchical equipment models where it is easy to locate and maintain. A model-based reasoning system produces from this model a decision-tree-like diagnostic logic that is converted to a hypertext document. Thus the location and maintenance of the knowledge is based on the context of the pieces of text given by a model

---

[35] Mainly due to the vague definition of the concept *hypertext*.

[36] See the references about the more recent hypertext research like [54, 68, 165, 237]. One should also note that rejecting the equivalence of the structure and the knowledge representation in hypertext rejects also one argument for the use of hypertext.

and the use of the advisory system takes place in a user-friendly hypertext environment (that would be tedious to use for creation and maintenance of the advisory systems).

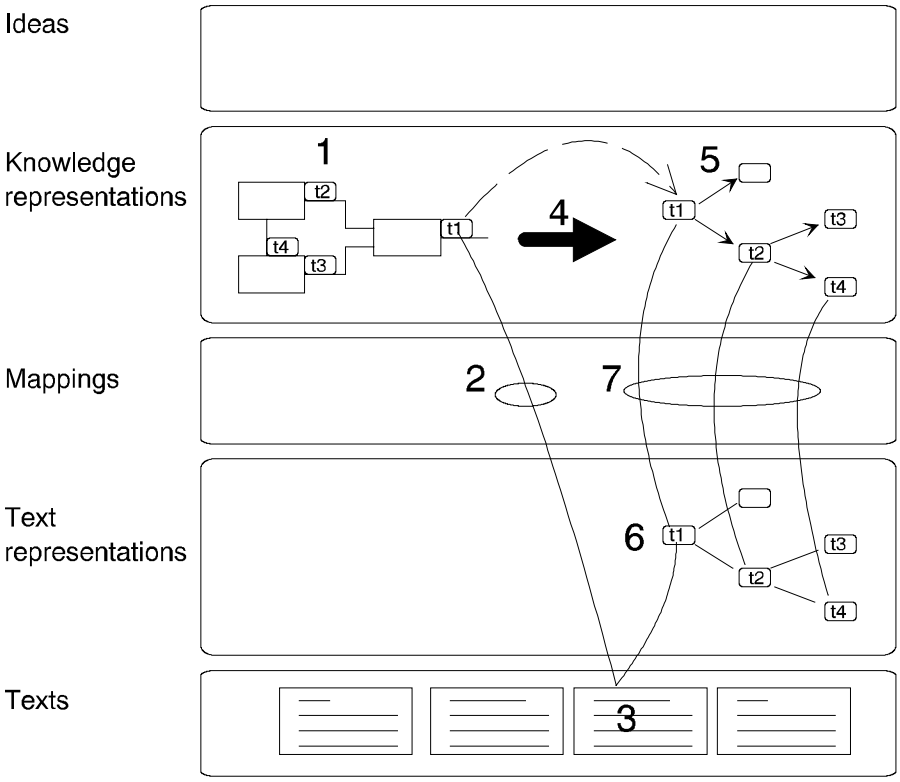

Figure 8. DMG: model-based hypertext generation.

Figure 8 displays the representations used in the DMG (Diagnostic Modelling and Generation) system. The equipment model (1) is represented as modules (large blocks), connections (lines between blocks), and tests (small, named blocks, t1..t4). The texts (3) describing the individual tests are linked (2) to the tests (t1). A reasoning component (4) produces a decision-tree-like diagnostic logic (5) that describes the test sequences as paths from the root (t1) to the faulty modules at the leafs according to the test results in a path (f.ex. t1 -> t2 -> t4 -> ...). This structure is converted to a hypertext network (6) utilising the one-to-one mapping (7) between the diagnostic logic (5) at the knowledge representation level and the structure of the hypertext (6) at the text representation level. The nodes of the resulting hypertext (t1) contain the texts (3) describing the tests.[37]

In this approach the creation of the advisory system utilises the separation of the text level and the KR level; the logic of the system is created and manipulated at the KR level independently from the preparation of the texts at the text level. For example, the chaotic character of the generation process (4) sometimes changes the structure of the optimised diagnostic logic (5) dramatically after small changes to the fault probabilities of the model

[37] Usually the texts and the hypertext structures have an one-to-one mapping. In this special case, the delivery environment enables several hypertext nodes to use the same text content, i.e., the mapping between the document structures and the texts is n-to-1. In addition, several nodes are used for describing a test.

(1). This changes the structure of the hypertext application in ways that are impractical to perform manually at all. The structure of the generated diagnostic logic is also used to pin-point shortages of the equipment model -- a feature utilised extensively during the knowledge acquisition process. Furthermore, the exhaustive knowledge representation manipulations and the delivery of the hypertext advisory systems can be performed in two different computer platforms.

4.4 Intelligent Databases

During recent years there has been an emphasis on integrating results from different areas of research to a system capable for storing and processing large amount of information and knowledge. These frameworks and unifying models include components from artificial intelligence (expert systems/knowledge-based systems methodology, logic programming, and object-oriented databases [37, 210, 250]), from database and storage technology (object-oriented, deductive, semantic, relational, and text databases [89, 118, 250]), from information management and retrieval (IR methodology, hypertext, multimedia, document management technologies, document image processing [40, 150], and optical character recognition [219]).

The names given for the models include Expert Database Systems (EDS) [109], Deductive Databases, Intelligent Information Systems (IIS), Intelligent Databases [165], Knowledge-Based Hypermedia Systems (KBHS) [159], and Intelligent Documentation Management (IDM) [150] (see also [26, 28, 176, 237]). Depending from the point of the view the text retrieval issues are either a by-product of the other capabilities or the main interest of the research. The approaches most relevant to this work are FORM and IDM. The FORM formal object representation model [165] is a unifying model that combines databases, object-orientation, expert systems, hypermedia, and information retrieval as an intelligent database. Gerald P. Michalski [150] describes the term Intelligent Document Management (IDM) as an extension to the term electronic imaging system where the related technologies in IDM include text management, information refining, complex documents, multimedia databases, network issues, forms processing, groupware, and hypertext.

The work under title intelligent databases contains various elements at the KR level as well as at the text representation level. They include highly interesting views from several points of view, but surprisingly, do not contribute as much to the area of this work as could be expected. With respect to the other approaches represented in previous sub-chapters, these approaches do not offer major improvements to the ability to integrate KR and text by text retrieval or by other means. A typical feature for these approaches is that they concentrate on the deductive capabilities of the KR level and pay less attention to the mappings and text representatives.

4.5 Evaluation of Alternative Approaches

From the point of view of our framework none of the text retrieval approaches analysed above provided sufficient representations and methodologies for all the three groups or levels of representations. The knowledge representations used in traditional library methods (i.e., subject classifications and single or multi-word index terms) are mapped manually to the texts, that is a highly elaborate, but widely used method. Use of single word terms contained in index phrases as the mapping from KR to text representations is a simple but also a very rough mappings method. Use of an advanced thesaurus gives means to represent KR-level concepts and their relations and improves search effectiveness [17, 122]. Unfortunately it requires an elaborate thesaurus construction process [69]. From the point of view of the framework the full-text approach -- including also vector space, probabilistic, and statistical models -- lacks knowledge representations and is highly string-dependent. Although these contemporary methods display great improvements in retrieval performance (measured in precision/recall) compared to the Boolean techniques [186, 187], they have not reached any major position on the mainstream markets dominated by the Boolean techniques [219].

From the point of view of our framework, the NLP approaches succeed well in combining the text representations and knowledge representations via text processing. The main drawbacks are the vast human effort needed for creation of the domain-dependent KR models and mappings as well as low processing speed. Thus the statistical evaluation of NLP-based methods are likely to suffer either from too small test sets due to poor efficiency or from poor coverage of the linguistic or domain knowledge.[38] This may be one of the causes for ignorance of NLP methods among the IR community [81]. Another potential cause is the underestimated importance of lexical variance due to the dominance of English that incorporates only simple variations in comparison with languages, such as Finnish [161] and Hebrew (or German [215]). So far the debate on usefulness of multi-word index terms and the usefulness of NLP in TR has not reached any final solutions [20, 35, 58, 188, 215], although some evidence exists that the systems incorporating NLP techniques are more effective than systems based on stochastic techniques alone [128]. Thus the use of natural language processing, and other complementary or alternative techniques, such as knowledge representation, or use of document structures is justified mainly by other aims (recall p. 20) the representations of domain's central concepts should be explicit, efficient manipulation of natural language requires automated tools, and tools are needed to make the IR systems user-friendly rather than user-hostile. NLP is expensive but still a much better choice than the use of manual indexing or abstracting when real-time processing of high volumes of text is needed. The successful automated abstracting products utilise NLP and

[38] If only 70% of sentences are correctly parsed, the use of syntactic structures in index term extraction is not likely to reach 100% precision or recall (without forgetting the other dimensions of the problem).

knowledge-based system methodologies, such as in AIDA [62] and in FLEXICON [71], or the ES approach used for automated keywording in JAKS [51, 136]. (See also articles of copier manual retrieval [141] and news categorisation systems, such as CONSTRUE [29] and SCISOR [101]).

The structural approaches, including hypertext, use simple and fast methods to map the knowledge and text representations. The academic ignorance of formal properties of texts is likely to be due to their restricted theoretical interest, but they are widely used in operational TRSs. The document structure approach suffers problems similar to traditional classifications, i.e., maintainability of the structure definitions used as the KR. In addition these approaches need still complementary NLP methods.

The hypertext approach goes even to one-to-one mappings between the structure of the documentation and the KRs that makes the interaction extremely easy with small systems but makes also the maintenance, creation, and browsing problems of large systems extremely hard. Experience shows that users find it hard to formulate their information need in form of a query [43] and the usefulness of a hypertext paradigm has been agreed on by the hypertext research community. Still hypertext systems have yet to prove themselves clearly superior over linear text in a wide variety of domains. "If anything linear text appears to have fared better in the few comparisons that have been carried out [237]." The recommendations about usage of hypermedia and other IR systems are [237]:

- If users know exactly what they are looking for (i.e., querying) then following links is not an effective way of finding the information and mediated retrieval is needed. In addition, structuring (menu hierarchies, etc.) helps on narrow down the topics of interest.
- If users do not know exactly what they are looking for (i.e., browsing) then following linear sequences is more effective than jumping around. Structuring and mediated retrieval may also help.

4.6 Discussion

Our framework described above emphasises use of knowledge representations primarily as a media for transferring conceptual knowledge that is inspected by browsing and navigating, and only secondary as a level of representation, to which and on which text content representations and user queries are transformed for comparisons. This view was adopted since the comparison of explicit users' queries with other representations entails naming the target before it can be searched for as well as due to the problems in selecting the comparison criteria and context for the comparison. As the knowledge representations are used merely for user interaction, they need not (and in fact, should not!) represent the complete meaning of a text as argued by some criticisers of knowledge-based text retrieval

systems [187, 188]. Their purpose is to represent the central concepts,[39] whatever they are considered to be.

The contemporary discussion on the relation of symbolic vs. connectionist methods in cognitive science, AI, and philosophy in general tries to understand the basic methods of human cognition as well as to find implications to areas, such as knowledge representation [152]. From this point of view the position of the framework described above is clear: the models should contain explicit, symbolic representations in order to enable information interchange between the authors, text indexers, and text readers. Without propositional structures as the media, the communication between them is impossible.[40] Instead, the mappings between these representations and the text representatives or texts can be based on any method that is able to link the texts or their representatives to the knowledge representations, such as neural network implementations. Of course, the presuppositions, explicit knowledge representations and texts / text representatives, give a rather symbolic basis for them, but at least at the level of implementation techniques the connectionistic approaches can be utilised.

[39] A common view to the purpose of knowledge representations in natural language text-retrieval is that the purpose of NLP is not to represent the full content of text in some KR formalism but rather extract the concepts that are important for text-retrieval purposes. (Conclusions of Concept Representation subcommittee at AAAI-91 Natural Language Text Retrieval Workshop, Anaheim, CA, July 15, 1991)

[40] This issue of the media by which the knowledge representations is transferred should not be mixed with the process where a reader interacts with the representation and matches it against the mental representations that may or may not be propositional.

5 TECHNICAL DOCUMENTATION

5.1 Purpose of Technical Documentation

The previous chapters have discussed text retrieval systems, their knowledge representations, and their representations in general. The purpose of this chapter is to focus the attention on technical documentation by introducing its special features and the underlying basis for use of KR components in TRSs in technical domains. Throughout this chapter the role of technical documentation will mainly be seen as a media for transferring knowledge that is needed for solving problems that are needed to solve for accomplishing the mission of the organisation.

This chapter starts by discussing the role of documentation as a means to transfer technical knowledge about large systems and continues by defining what is meant by technical documentation. The next sub-chapter describes the problem-oriented nature of text retrieval in technical domains and presents diagrams for describing TRS performance from this point of view. The last sub-chapter discusses the economical criteria for use of a TRS in technical domains, i.e., the relation of the costs and benefits of a TRS during its life-cycle.

5.1.1 Documentation -- Corporate Knowledge

The centrality of text in human communication is commonly accepted [131, p. 287]. Corporate manuals and documentation contain a large proportion of the fundamental knowledge of an organisation. This knowledge is often crucial to the efficiency of the organisation and is the keystone to profitability and providing the competitive edge. The quantity of technical documentation, especially software documentation, produced daily is surprisingly high. Besides Bibles and telephone books, one of the largest world-wide publishing efforts is the documentation of computer programs [140 p. 44]. This activity receives relatively little attention from the general public or even from the computer-technology community.

Where scientific activity consumes and produces information in verbal form technical work produces mainly physical products and processes and documentation is only an incomplete by-product. It generally assumes a considerable knowledge of what the physical product is. Those unacquainted with the actual product development require some intervention to supplement and interpret the information contained in the documentation. Thus, technical documentation is often most useful only when the author is directly available to explain and supplement its content. [6, pp. 2-5]

5.1.2 Text vs. KR as a Medium

Text can be a very effective way of storing knowledge since it is, in general, easy to prepare and straightforward to understand. However, the richness of natural language can lead to ambiguities and inconsistencies and authors often require considerable language skills to ensure that the text will clearly convey to the reader the meaning intended. [147] Formal knowledge representations can avoid some of the inherent problems of natural language; the syntax and semantics of the representations can be defined unambiguously based on deterministic, context free interpretation procedures. The strengths of formal representations in knowledge transfer are mainly based on their clarity and computers ability to enforce consistency rules as well as to re-organise and restructure them. The communication from a human or a system to a human via formal representations requires extra effort from the human in learning the formal language. Thus the major issue seems to be the trade-of between minimising the cognitive overload caused by languages with sufficient generality and expressive power.

It can be stated that in cases where the knowledge to be transmitted is formal, structured, modular, and homogenous containing lots of similarities and regularities, the advantages of the use of formal representations is advantageous. Instead, use of natural language is preferable if the knowledge is fuzzy and complex containing mainly unstructured and unique features. On one hand, it seems that the ratio of the size of the meta language needed for representing the knowledge representation formalism and the size of the representations indicates the usefulness of the representations.

From the first point of view the use of subject classifications/headings or index terms is very efficient. The meta language states only that a classification system contains classifications that may have sub-classifications treated as a specification of the more general one. From the semantic clarity point of view the subject classifications are somewhat vague. The criteria for dividing a classification to sub-classifications varies from geographic location and language to taxonomy of disciplines. Thus human interpretation is needed during the use of the representation and computerised restructuring can take place only in a very general level. The meta-model of a standardised thesaurus is somewhat more elaborate differentiating the semantics of relations, such as *broader than* and *related terms*, but still vague due to the requirement of generality and domain independence. If a hypertext is treated as a semantic network describing domain knowledge, the node/page/card - edge/link meta model is efficient and ambiguous like the subject classifications and thesauri. If the meta language is elaborated by defining types for nodes and links, the cognitive load needed for users to adapt the system increase, but the semantics of the representations can be clarified. The knowledge representations used in NLP systems on one hand, search for economical representations for representing huge volumes of knowledge and on the other, try to include also all exceptions in the representation. Thus the

meta-language and the representations are elaborate and optimised for representing linguistic knowledge.

5.2 Technical Documentation

5.2.1 Documentation and Product Life-Span

By technical documentation [174] we mean all the documents involved in the whole life-span of technical products, from design to after-sales, as well as other documentation of technical organisations. The nature of documentation varies according to the products as well as to the state of the product's life-span. In this work special attention is paid on in-house and customer documentations describing large products, product families, or systems, their operation, and use. The product family aspect here refers to the wide area of problems related with managing information about different product versions and configurations, i.e., how to include the appropriate versions and only those versions needed to describe a system as one unity of documentation.

The documentation available before the product design includes documentation of previous products, information about existing products used in the development, material provided by vendors and customers, standards, and text books. Most of this material is unfortunately in paper form although the previous designs may be stored in design databases. During the design process a large number of documents are produced; textual specifications, program code, schemes, layout diagrams, simulation models as well as other non-textual material. In this phase majority of the documentation may be in non-textual format within CAD databases. All this dynamically changing information is typically collected into design databases where the current versions of the information are available for all who are involved in the design process.

Before a new product reaches the production phase several additional documents will have been produced. Production documents include textual and graphical information about the production process for humans and computer programs and parameter information for numerically controlled (NC) machines. Test documentation defines the test procedures and criteria for acceptance of the product as well as the test programs. Also in this phase the non-textual information may form a major part of the documentation.

The marketing is involved in the process all the time. First documents are used for product marketing even before the product design process has really been started. More detailed technical information is needed for technical support organisations of the marketing units during the sales of advanced products. The marketing organisations often utilise the databases of the design organisation in form of product break-down structures and text databases. Customer documentation is typically the most elegant part of the product documentation. Its' size varies from an one-page leaflet to several hundreds of thousands of pages technical information delivered with a complex product, such as an aeroplane. Also

the maintenance organisations need documentation for efficient field service, mainly in form of textual (paper) documents although expert systems and hypertext systems are used to some degree.

5.2.2 Language and Structure of Documents

A document has no value as such, like novels, but as a mean to transfer a certain amount of knowledge to the reader. Thus the authors attempt to communicate as much information as possible in as little space as possible without sacrificing too much clarity. Both structure and content is more condensed and rigidly organised than in average documentation. Language usage is more dense in nature than normal prose. The documents use a limited vocabulary and somewhat limited grammar. In a compact documentation there is no need to express ideas with alternative ways which might even sacrifice clarity. Technical documentation may even support use of otherwise non-standard constructions for the sake of brevity. In addition to limited common terminology technical documentation uses a great number of domain-specific terminology [108]. According to our observations in a software oriented domain [211] some 10% of unique terms -- about 4000 out of 45 000 terms in a dictionary -- are domain-specific terms. The characteristics include a large number of multiple-noun compound terms as well as a high frequency of abbreviations used for them.

The aim to minimise the number of pages used for documentation occurs also in the form of extensive use of lists, tables, figures, and other compact means of representation. These representations are typically faster to produce, clearer to read, and replace a much larger amount of prose-oriented running text [244]. Some of the complex and recursively embedding structures used in technical documentation cause problems for natural language processing systems. Consider, for example, list structures and figures containing words or phrases. If a list structure is treated simply in text-only format, as a sequence of words where the items are concatenated together, it will typically produce a non-grammatical 'sentence' containing a sequence of phrases or words. This means that the techniques used must be able to utilise mark-up structures and further to manipulate also incomplete sentences. The mark-up used should also represent explicitly the structures of the documents. Implicit structures, such as tabulated lists, are in some cases difficult to recognise without errors.

Technical documentation of small systems do not differ from documentation of other domains as much as documentations of large systems. Large documentation containing tens of thousands of pages need higher level of organisation. A common way to organise manuals is a set of collections of similar pieces of documentations. Especially in software documentation the reference manuals are typically organised according to fixed structures for groups of entities that are described in a strictly controlled format containing a set of fields similar to database record fields. Consider, for example, commands in UNIX or CommonLisp manuals, functionalities of a text processing system as well as operations of a

process control station follow this paradigm. The fields of a command reference manual may include fields, such as name of the command, arguments, options, description, examples of use, known errors, and "see also" or references to other documentation.

5.2.3 Design Databases and Documents

In some cases the creation of documentation is integrated to the design process. Some documents are even generated by programs that format (parts of) documents based on information of design databases. These documents have all a specific structure and can be described as records of a text database or even as "instances of a document class". There are obvious similarities between these documents and the reference records in library information systems. Each of the fields is associated with a predefined domain of content and expectations about the contents that the manipulation of a database may utilise. For example, the text in field "see also" contains references to other command descriptions that can be used as a basis for automatic hypertext link creation.

Technical organisations are using database technologies for the management of the product databases as described earlier. In addition to the computer aided design and software engineering systems (CAD, CASE) many organisations use formal design languages (SDL, VHDL, etc.) and database management systems based on relational as well as hierarchical and network datamodels. A typical database application describes product break-down structures elaborating each product to sub-products and parts up to the required level of detail. This hierarchical structure is used to organise several kinds of data related to the products, including documentation of the products, sub-products, and parts. In addition to this information the design databases describe various relations between the products and other design entities. Thus there exists lots of information that can be utilised for text retrieval purposes[41] although it needs some further refinement with respect to the format and level of detail used.

5.3 Text retrieval Process in Technical Domains

5.3.1 Problem-Oriented Text retrieval

It can be stated that text retrieval problems may be easier in technical domains than the problems in general. First of all, the domain is typically somewhat restricted and thus the information needed for describing the semantics of the domain is limited, sometimes formalised and even available in electronic form in databases! From the NLP utilisation point of view technical documentation has the advantages and disadvantages described earlier. In general, technical documentation as such is more oriented towards structure-oriented approaches in TR than texts in other application areas.

---

[41] As well as for knowledge base generation. See f.ex. [154]. The interchange of knowledge in general is discussed in [156].

In this work the focus is on text retrieval by computerised methods which in technical domains implies that most of the documentation is produced by the organisation itself, by vendors, or is extracted from some external databases. With the contemporary state of the business this means that textbooks, journals, and other printed material are somewhat neglected and the focus is on manuals, technical reports, and similar material. This kind of material is mainly used for specific work activities and less on keeping informed, professional development, or other general activities.

In technical domains [29, 174] (as well as in offices [31]) the use of documentation is characterised by task orientation [166]. For example, in software engineering domains it can be stated that, "Nobody reads documentation. [181]" -- "except as a last alternative [80]". Manuals should not just describe the features of a system, they should help people get things done. The so called software-oriented manuals do not help in finding the correct command and the index is of no help [29]. Menu-oriented manuals do not help much either. What is needed is task-oriented manuals that give guidance for each task. They provide procedures for all the things one might want to accomplish with the system. [181]

The use of external textual material is often secondary to consulting human experts or use of personal data files. In some studies [174] a network of knowledgeable co-workers is seen as a secondary source of information filling up the gap left by poor technical documentation. According to other sources [204, pp. 37, 40] engineers tend to prefer another person as a source because they can select just the information needed to solve a practical problem at hand and can present the information in a suitable way. The sources used most frequently by aeronautical engineers and scientists to solve technical problems were personal knowledge, informal discussions with colleagues, discussions with experts in the organisation, discussions with supervisors, and only after them, journals and conference/meeting papers, handbooks and standards, government technical reports etc. [171]. In some other sources [6, p. 40] information is transferred in technology primarily through personal contact. In acquisition of information the engineers prefer personal files, borrowing from colleagues, and personal library searches to the use of technical abstracts or search by library assistance [6 pp. 83, 92].

The typical text retrieval process in technical[42] and in commercial environments can also be characterised as problem-oriented.[43] The process starts when (and only if) someone has a problem to solve or a question to answer but has no relevant information in hand or cannot reach anyone who could answer the question within the given time constraints. Typically the need for information is not defined easily with a couple of words but can be described verbally with some sentences. In technical domains the users can be unaware of the specific

[42] For scientists and engineers [113] the principal purpose of most readings involves specific work activities - for technical reports in 90% of cases and for other activities typically 65-70% of cases.

[43] From the point of view of information needs and uses [47] the view adapted here is oriented towards qualitative research and "Alternative" paradigm and is closest to the approaches like the anomalous states of knowledge approach.

terms that should be used to define the information need due to the vast amount of domain-specific terms or the fact that the users are unable to name the concept(s) rather than to describe some of its' required properties or relations to other concepts. It may also be that the users know the solution (or appropriate search terms [220]) but have forgotten it. The users may have found the needed piece of text earlier or have glanced through a text or even several pieces of texts that contain the answer. In some other cases the users may also be unaware if the required information exits at all in the system.

The problem-oriented TR process may start with some heuristic methods to find some text that contains the required information. If the users have seen the answer, they typically try to repeat the steps of the previous interaction with the retrieval system. The context of the previous session may be helpful, for example, the topics of the last session or the dates of the documents read can be useful. In general, the associations and structural relations between pieces of information are utilised exhaustively. If the users have no pre-knowledge of the existence of the information, they are less attracted to start a retrieval process at all than to ask for a co-worker for advice, if possible.

The retrieval process continues until the user finds the answer to the problem. Typically one of many possible pieces of information is sufficient to solve the problem even if more than one relevant text exist in the system. The situation, where a user wants to find all texts relevant to some topic is not as typical for technical or commercial environments as it is for example for text retrieval in academic environments. Thus the retrieval process goes on until the user has found an answer to the question or becomes confirmed that it does not exist in the system or runs out of time. This view is supported by other related work. Cleverdon [34] recollects only four occasions when the end-users were trying to obtain 100% recall; the vast majority of users required a few relevant papers. Lanz [125] found out in his experiments with scientists and engineers that only a part of the relevant documents retrieved in online searched were subsequently used; the engineers never used more that eight papers even though a hundred relevant citations were found. According to Kristensen and Järvelin [122] journalists also seem to work in similar fashion.

5.3.2 Cost and Result of the Process

No text retrieval takes place in industrial or commercial environments if the value of the text retrieved is less than the effort needed for the process. If the probability to retrieve the information is small the threshold for starting the TR process grows correspondingly. The threshold for the use of an TR system is determined not only based on the value of the information, but also based on the cost-effectiveness of the alternative methods to solve the problem, to consult an expert inside or outside the organisation.[44] The cost of the process

[44] Allen [6, pp. 182-191] observed engineers' use of nine alternative information channels by measuring the frequency and order of channel access. The engineers decision was mainly based upon the criterion of least average rate of probable work, i.e., a reminiscent of Zipf's "Law of Least Effort".

can be measured in several ways. Usually the monetary costs are the most important criteria for all activities performed in commercial environments. In TR this criteria is often equivalent to the time spent on the process because of the costs are typically linearly dependent on the time used for the process, especially for the salary of the user and the costs of communication lines and equipment. In addition, the constrains that force the process to stop before the answer is found are typically time constraints. Thus the time used for the TR process is in most cases the best measure for the costs of the process.

In problem-oriented text retrieval the number of found relevant texts is often irrelevant in technical domains.[45,46] There are two main alternatives for the result of a TR process; either the solution was found or not. The first alternative is clear, but when we are evaluating the performance of a TR system the second one requires some further inspection. In cases where no answer for the question exists in the system, the failure to retrieve it is inevitable and should not be considered as a failure of the system or the methods used. In this case an ideal system would minimise the effort that users spend on the retrieval process in order to minimise the costs of the process. In other words, the system should be *transparent* in such a way, that the user are able, as soon as possible, to determine whether the required information exists or not.

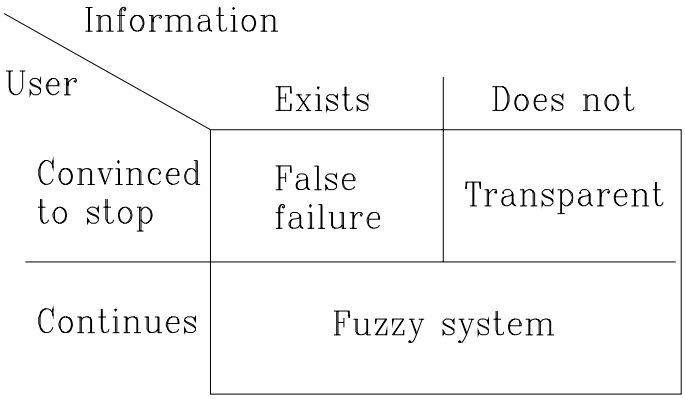

Figure 9. Possible results of a TR process, where no solution was found.

The users are faced with a kind of stopping problem; when should one stop the process if one has not found the information so far? The Figure 9 illustrates this situation. On one hand, if the required information does not exist, the user should be confirmed to stop as soon as possible. On the other, the system should not encourage the user to stop when the required information exists. Not at least if the user is able to retrieve it within the time

[45] Or as stated by Cleverdon [34], if more than n citations are unwanted, they are considered to be irrelevant. In case of problem-oriented TR n is very small, even 1 when the user needs only an answer to the question.

[46] This implies also that the use of feed-back methods is often less appropriate than in other domains; if the text-retrieval process ends when the first positive example is found, the feed-back cannot utilise the differences between irrelevant and relevant texts. The difference between "more and less irrelevant" texts is still available.

constraints.[47] If the system makes the users uncertain about the existence of the answer, the users are likely to spend lots of effort on trying to find the information until they find the further work no longer to be cost effective, run out of time, or meet some other constraint. This kind of situation is most unpleasant since the users may have to repeat the process in some later day if the same problem reoccurs.

5.3.3 Effort/Result Diagrams

In summary the problem-oriented TR process assumes that

- a need of information is in form of a problem or a question,
- a TR system is one of the competitive methods to find the solution,
- retrieval of only one of potentially many solutions is needed,
- the solution may or may not exist in the TR system,
- the criteria for the effectiveness of TR are the result and the cost,
- there may be constraints on the effort (cost, time used), and
- searching for non-existent information wastes effort and should be avoided.

During the TR process the users are unaware if the result exists or at least unaware if they are able to find it within a reasonable time.[48] Sooner or later they stop the process successfully or fail. The success can be either that the solution is found or that a correct decision to stop the process is done, when the information does not exist in the system. The cases, where the users are confirmed to stop, because they believe it does not exist, or fail to find the required information within the available (time) constraints are considered to be failures. The later of the failures depends, of course, on the constraints of each of the cases.

The Figure 10 represents a diagram where both the information about the cost of the process and the result of the process are summarised into an useful form. The horizontal axis of this diagram represents the effort spent on the retrieval process before stopping and the vertical axis represents the cumulative percentage of processes in each of the cases A, B, C, and D, at this point of effort spent, by the distance between the border lines. Thus the percentage of C decreases from 100% to 0% in the infinite and the percentages of the others increase. In the Figure 10 the portion A represents the cumulative percentage of cases where the solution was found. The portion B represents a cumulative sum of the cases where the users are able to recognise, that the required information does not exist and stop the process. Sum of these two cases, A+B is considered to be the percentage of successful TR processes. The portion marked with C represents the processes, that are still going on after the effort shown in the horizontal scale is spent. This portion decreases from 100% to 0% when more

---

[47] If the user is not able to retrieve the required answer within the time constraints due to poor performance of the system, the system should anyhow encourage him to stop in order not to spend extra time on the process! Even if the user knows that the required information exists in the system or has indeed retrieved it earlier!! This proposition can, of course, be argued.

[48] A most relevant feature ignored by approaches like [213].

effort is used. The portion D displays the percentage of cases, where the user stops the process before the existing information is found, either because the users are confirmed that it does not exist or consider it to be waste of effort to go on. The sum of cases C+D is considered to be the percentage of failures when the limit of maximum available effort is spent.

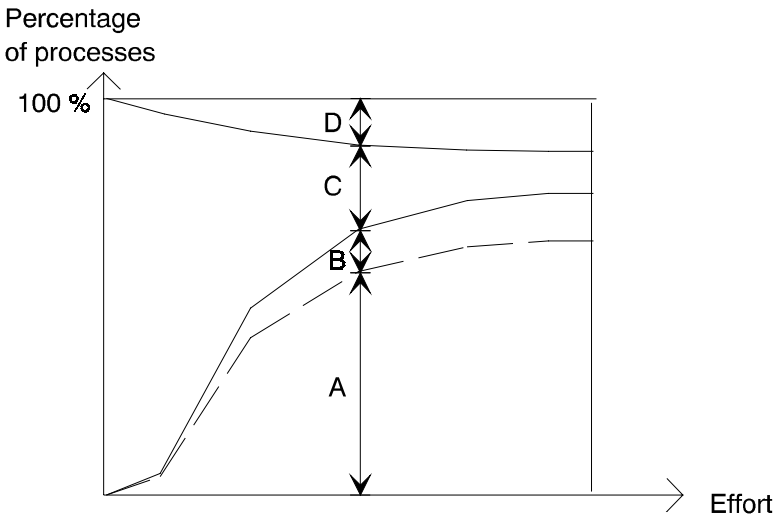

Figure 10. A diagram describing the performance of an TR system.

The curve D and curve B are dependent on the percentage of the cases, where the solution does not exist (i.e., B+D). If the required information exists always, the set of curves reduces to the curve A. This corresponds to the situation supposed in P/R measurements. Thus the curve D is also relevant for the determination of the system performance, but the curve between A and B can be ignored as uninteresting if the percentage of cases with no answers is fixed.

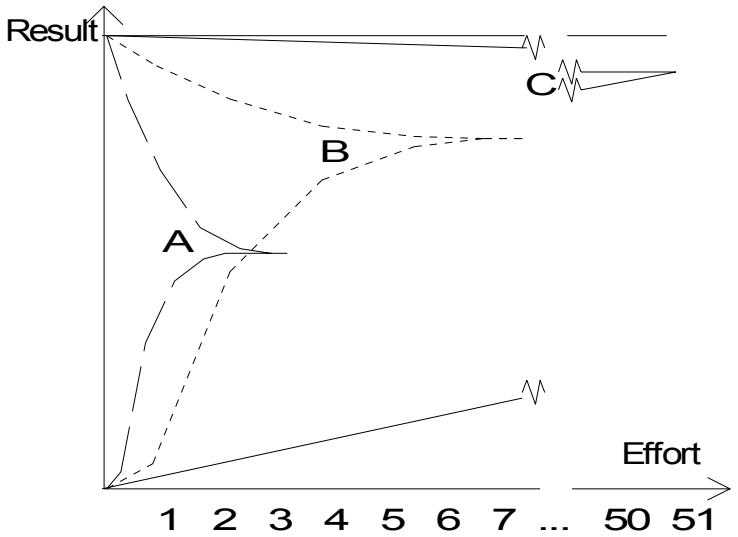

Figure 11. Effort / Result diagrams of alternative TR methods.

The effort/result diagrams do not only capture the two most relevant dimensions of the problem-oriented text retrieval process in a descriptive manner -- they are also applicable for comparisons between separate text retrieval methods. For example, a comparison of a query-based IR system with a hypertext system is rather difficult with other means. Figure 11 displays an effort/result diagram describing three alternative systems/methods (A, B, and C) in a hypothetical controlled environment. The methods used might be a statistical method with relevance feed-back, a traditional keyword search, and reading through the documentation. The results display clearly, that the method A enables the user to find the solution fast in many cases, but also misses the needed information often. The method C is very slow, but is fairly sure to find the information, if it exists. The method B seems to be the most useful in cases, where there are some constraints to the effort although is not as fast as the method A. Method B also leaves the user uncertain about the existence of the information more often than the method A, i.e. the users are convinced to stop later than with method A, after spending up to 7 units of effort / time instead of 3 units that is at most spent with method A. In applications, such as stock exchange information systems, where the constraints are high, the method A would seem to be most useful. Usually the method B would be chosen due to the ability to retrieve required information in most cases. In applications, such as legal case retrieval systems, the most important goal is not to miss the critical piece of information. Thus the method C is also needed, at least in a combination with some of the others. These quantities, summarised in the diagram, are the system-specific information relevant for the evaluation of text retrieval systems in problem-oriented text retrieval. One should still remember that they are only a tool for representing the essential information that has to be gathered with extensive field tests, in a controlled environment.

5.4 Selecting TR Methods

5.4.1 Contribution vs. Costs

The decision to use a document management or a text retrieval system in a technical organisation is based on economical justification; the use of the system must pay back the investments within a reasonable time. As a result, cost is always the largest obstacle impending the acceptance of these or any other new technologies [83, 120]. Despite benefits of increased productivity and new opportunity, the cost of reaching new capabilities is still the key of justification. The dilemma is that costs and benefits are always measured within the context of existing business models. Sometimes the effect of a new technology changes the model so dramatically that the old rules of measuring costs and benefits do no longer apply. In this case we are not dealing only with a new technology paradigm, but also a new business paradigm. This is the fundamental issue facing evaluators of text retrieval systems.

The question is, how to evaluate the costs and benefits of a technology which has little or no precedent in the organisation. [120, 121]

One of the proposed criteria for success of text retrieval technology is its' affect to organisation's success criteria, i.e., the affect to the means by which the organisation measures its effectiveness [120].[49] In medical industries this criteria may be time-to-market, in law the ability to find the vital briefing, etc. The key is to find the criteria at the enterprise level since strategic investment apart from the key success criteria may minimise costs but produces bad result. The advantages of new opportunity gained by utilisation of new information system technology can be measured as the new *contribution level* that is the systems "identifiable and tangible contribution (positive or negative) to an organisation's key success criteria. [120]" This approach compares the total cost of the existing system, which includes lost opportunity through increased risk (i.e., costs caused by the inability of the current system to support the key success criteria), with the real benefits of an alternative system; increased opportunity through reduced risk. In some cases the lost opportunity may be crucial for the mission and existence of the organisation. Consider, for example, the knowledge worker searching for information in corporate documents, the scientists scanning research reports, or the lawyer looking for a crucial evidence.[50] What costs are involved in uncovering that buried knowledge? As stated by Spayd [207]: "The cost of knowing is sometimes great. The cost of not knowing however, is sometimes fatal."

Document management and text retrieval systems do not increase productivity only by automating manual tasks. Re-engineering information systems also redefines the process and collapses the business cycle. Time required for tasks which were unfeasible and impractical can be reduced dramatically. This allows for research and decision support that would not have even been considered before automation. The impact that collapsing the business cycle has on the bottom line can be easily measured in a number of ways; faster and higher quality decision making, shortened research and development time, advantageous position on market place, and reduced sales cycles. Thus research on new work practices is as important as research on new products [25]. [120]

The evaluation of text retrieval systems based on contribution level or increased productivity requires qualitative estimations and produces best overall view to the advantages gained by the organisation. In some cases electronic information storage, delivery, and text retrieval systems can be justified even with conventional cost reduction basis [80]. The cost savings may take place in form of reduced labour costs (printing, copying, mailing), saved material (paper, folders, letters), reduced external services (drawing, copying, copier maintenance, paper garbage disposal, mail, insurance, taxes),

---

[49] There are also on one had, more general cost/benefit oriented approaches and on the other, approaches oriented more towards effectiveness of text-retrieval (e.g., work of Lancaster and [47]).

[50] According to a survey by Dun and Bradstreet in France, 73% of lawyers said that the costs of information is not important [52].

saved rents (office rooms, storage, archives), avoided equipment purchases (copier machines, air conditioning). Use of electronic documentation also reduces costs for updating documentation, shortens delivery times, and even improves working environment by avoiding paper mountains and paper dust.

Analogously to the decision to use text retrieval technologies the selection of techniques is application-dependent. Typically there exists some primary criteria that must be fulfilled before the cost efficiency of the system can be optimised. In between these two there are alternatives that compromise between different factors and costs of the configuration. These application-specific constraints can include:

- the required effectiveness and efficiency of text retrieval process concerning accuracy (precision, recall), response times, quality of the response etc.
- the size of text database influencing also precision requirements, storage media capacity, and production costs,
- the time, material, techniques, and human resources needed for producing a retrieval system, influencing the choice of techniques and the media,
- the HW/SW platforms, their availability and tailorability,
- etc.

5.4.2 Costs of a Text retrieval System

The costs of text retrieval systems are easier to estimate than the benefits [121]. They include HW and SW costs as well as labour cost. Spayd [207] divides the costs into two basic factors: costs of system set-up and the costs of actually retrieving information. According to Spayd the system set-up costs are explicit whereas the retrieval costs tend to be hidden. Typically the retrieval costs are much higher than the set-up costs. Further, increasing the system set-up costs usually reduce the retrieval costs. Unfortunately these two factors are not balanced optimally due to the explicit nature of the set-up costs; they are often minimised and the total costs are increased due to the increase of hidden retrieval costs.

For our purposes it is convenient to elaborate the set-up costs to preparation costs for the system use in a new domain and to processing costs needed for including new texts to the system. This is important since the preparation costs are rather fixed costs paid only once whereas the processing costs vary according to the amount of texts processed. In addition, the processing costs may be either explicit or hidden costs. In organisations that produce regularly large quantities of documentation the processing costs can be separated explicitly. Instead, if the documentation is produced as a by-product or some external texts are fed into a TR system, the processing costs are typically hidden. Thus the three main cost factors according to the phases of system usage are:

- Preparations
  Language and domain-dependent tailoring including HW/SW costs for system set-up, labour costs for construction of domain-specific lexicons, system synonym list, thesauri, indexing rules, hierarchies to group or categorise concepts, domain models, knowledge bases, converters and interfaces to other information systems, etc.
- Processing
  Computer and communication capacity and human costs spent on composing, converting, transmitting, and processing a text, including indexing, classification, NLP, statistical analysis, creation of hypertext links, or other operations needed for each document as well as operations performed for the whole documentation, such as global statistical analysis, linkage of documents to knowledge bases, conversions, and production of delivery media.
- Retrieval
  Computer and communication capacity and especially human costs spent on composing (and recomposing) requests, translation of requests to the syntax of the TR system, construction of request macros, glancing through the returned texts, searching for adequate index terms or classifications, navigation of hypertext networks or domain models, etc.

From the cost optimisation point of view this division should be used for comparing orders of magnitudes of investments on different areas. The investment made for preparations is available for processing all the documents and similarly the once processed documents are available for all users.

> Let us, for example, suppose that there are a hundred of users retrieving documents from a TR system. If we can reduce the time a user spends annually on text retrieval from four weeks to three by increasing the (annually) text processing effort from a week to two months, we still gain huge savings (hundred user weeks against seven processing weeks).

Similarly, investment on preparations may give several hundreds of percents return on investment.

> To continue the example, if the improvement on retrieval was gained by indexing documents manually (i.e., improving processing), we may further save costs by purchasing an automatic indexing system (i.e., investing on preparations).

In general these calculations are somewhat application-dependent; if the authors index the documents manually, we have some hidden costs, whereas a centralised indexing (manual or automatic) produces explicit costs.

Costs of applying a TR system depends thus on one hand, on the performance of the system during processing and retrieval and on the other, on the tailorability of the system; the retrieval costs can be minimised by increasing the processing and preparation costs, and

the preparation costs are dependent on the tailorability of the system. In a large scale system it should be possible to tailor the system to fit the organisational, functional, and economical constraints of the organisation in question. The configuration of a system should be modifiable and the domain-specific parts of the system must be fast to build and to integrate into the system. The larger the documentation is, the more users there, are, and the more important the efficiency of TR is, the more important it is to provide the users with some overall structure of the domain. Unfortunately the preparation costs of classification and modelling systems have been rather high. The smaller the tailoring costs are, the smaller applications are able to utilise the preparations cost-efficiently. As the result the retrieval costs and also the overall costs of text retrieval are minimised.

6 DOMAIN MODELLING IN TECHNICAL DOMAINS

6.1 Special Features

This first sub-chapter summarises the special features of technical documentation influencing the domain modelling process. It includes features presented in previous chapters and presents some new ones. The next subchapter compares the traditional thesaurus construction approach (cf. pp. 18-) with the special features presented here. The major problems of it and other domain modelling approaches are analysed. The last sub-chapter summarises the requirements for KR used in domain modelling in technical documentation retrieval systems based on the previous chapters and the requirements presented in this chapter. It also summarises the problems of alternative approaches.

6.1.1 Artifact Warrant Principle

One of the characteristics of technical documentation described earlier is the frequent use of special, *domain-dependent terminology*. Technical documentations -- especially documentations of large products and systems (LPS) -- contain large numbers of new proper names, compound terms, and acronyms. This is because product development not only creates products but also *new concepts* and (usually) well-defined terminology to name -- to identify and to describe -- the new concepts and design entities. The management of a large system without unique naming for the parts of the system is extremely difficult -- although naming conventions may sometimes be more oriented towards easy use by computers rather than by human beings. This means that technical documentation includes not only general technical terminology and terminology of specific technical domains or sub-domains, but also -- and especially -- terminology used by individual organisations for describing specific products or systems. For the management of the former, we can possibly use external technology-area-dependent thesauri whereas the organisation- or product-specific terminology must be managed by the organisation itself. It is the organisation-dependent terminology which is now in our focus since vast majority of domain-specific terminology is exactly that (cf. p. 61) and it tends to change more frequently than the terminology established for a broader domain.

The system- or product-dependent concepts described in technical (LPS) documentation refer to newly created artifacts and artificial entities defined in design databases and named by newly created terminology. This has major influence on the various aspects of domain modelling process in technical domains making it inherently different from the process in other domains. As a result this thesis states that

> major parts of the domain model (thesauri etc.) and parts of the information used for mapping the model against texts should be derived from the design databases which contain the most accurate and consistent description of the product or the system in hand.

This principle is later referred to as the *artifact warrant* principle (AWP), in contrast to the user warrant and the literary warrant principles described in the previous chapters (cf. pp. 18-). Some of the special features behind the artifact warrant principle are rather obvious in typical technical domains whereas some are merely differences of degree. Still, most of them can be seen as implications of the artificial nature of the domain in hand. The special features discussed below include rather clear separation of terminology from the domain model, ability to access the definitions of the concepts in electronic form, rather clear mappings between the concepts and terms or document structures, and the need for version and configuration management.

The *separation of terminology from the domain model* can be observed on the basis of the model for levels of representation presented earlier (cf. pp. 29-). In the model the terms are at the text representation level and the domain models at the KR level. This separation of representations is advantageous in a TRS if we can easily identify concepts behind the terms although they may have ambiguous representations at the text representation level. The use of the KR-level representations would not be as advantageous if the definition of terms were mainly based on their usage. In the case of artifacts and artificial systems -- such as large technical products and systems -- the later case may even be impossible. The definition of system-specific terms cannot be based on their typical patterns of use before they appear in the documentation. If they have just been made up, only their creator (the designer) is aware of them and of their meaning, the concepts they refer to. In these cases we can identify the design entities and their representations in design databases as the concepts the system-specific terms refer to. Even the various levels of abstraction included in a system design may be formally defined and stored in design databases.[51] It should be noticed, however, that when a system or a product is described or discussed by people other than it designers, the portion of system-specific concepts and terms is likely to decrease to a large extent.

As the concepts referred to by the terms are typically *defined by the databases,* we have access to them. This means that the (definitions of the) concepts, their attributes, and their relations are *available in electronic form*. Further, the semantics of these representations is often specified to a high degree according to some definition language or database structure. This type of formally defined semantics is typically used in describing the properties, behaviour, functionalities, and relations of design entities. By extracting this information, it

---

[51] Instead, humans´ intuition about these concepts is typically influenced by incomplete information and previous associations. This does not influence the existence of the reality but human perception of it.

is possible to construct KR models describing the concepts and their relations with a high degree of accuracy and precision (i.e., *semantic homogeneity*), taking into account the special features and demands of the domain.

The next issue is the definition of the *mappings between the design concepts and text representations* (TR, e.g., terms and text structures). As described earlier, technical (LPS) domains tend to have methods to formalise and manage the use of the special terminology, i.e., define formally the mapping between the design concepts and terms. This is often established by associating a concept with a proper name -- often a new compound word -- that identifies the concept. In addition, an abbreviation may be created on the basis of the compound word or a combination of an abbreviation (describing a part of the compound word) and a word or words. Further, each design concept may be associated with identifiers, such as part codes, database surrogates, and the like. This means that a design concept may be mapped only to one or few terms or term-like identifiers. The same term or identifier may still be used to refer to some other concept in another context, thus causing ambiguity. Whether this happens depends on the terminology and identifier management policies of the technical organisation in question. In general, the relation between design concepts and terms is limited, constrained and controlled to a much higher degree than what is typical in other domains.

The *relation of concepts and text structures* can also be rather well defined in technical (LPS) documentation, as described earlier (see pages 47 and 62). This means that in many organisations documents may be located differently based on the central concepts they describe. The documentation, project management, and design policies may include a phase, where product codes, project names, other identifiers, and special tags are added into the documents by manual operations or by automated tools. Documents may also be managed by the same management systems as the products, enabling formal specification of the relation between the documents and the design objects. In general, in-house document production with standardisation policies enables, and often leads to, easier mapping between design concepts and text structures.

Very often documentation is the last phase in system development. In worst cases, major parts of the documentation of a large system is not available until the day when the system is ready. In some cases the documentation is delivered document by document only after a large system or product is delivered. This means that the domain modelling process may not have access to major parts of the documentation. Therefore, it should be possible to perform the modelling without the documentation.

Product families of almost similar products or systems are typical to technical organisations. It is natural that the models representing products in a product family are close to each other on a very general level. Also on the more detailed level parts of the products and their documentation are similar. The situation with different versions of a product is analogous. If the users with a specific product -- and a specific version of it --

want to retrieve texts related to a problem they have, they are most likely to be confused if the product in hand and the domain model mismatch. The mismatch may be on the level of missing or additional concepts included in the documentation and/or domain model or in wrong/additional/missing properties of the concepts or relations between them. All these may be caused by use of the same domain model for all products or systems in a product family or because of a version mismatch between the product, documentation, and domain model.

The fast pace of version development and product configuring is an additional problem for technical (LPS) domain modelling. New product configurations can be created "on the fly". Similarly, the set of product variants (possible versions) may be defined in the design databases implicit -- as a set of features, with each having a set of alternative options that together define a new product variant. This means that both the domain modelling and the documentation construction are forced either to be inaccurate or to change dynamically as fast as new product or system variants, versions, and configurations appear. Further, the creation of the domain model should not require much of the valuable time of the experts. In many organisations, the documentations are automatically compiled of small units based on the system configuration and some parts of documentation may even be generated automatically from design databases. If the document construction is automated, the domain model construction and the mapping between the domain model and the texts is the bottle-neck in the process. In some cases even the use of an existing domain model of one product of a product family and incorporation of the differences between the two products or versions may be too large or time-consuming an effort to be performed manually. Thus the creation of a domain model for a large system or product needs some degree of automation -- if not for all product variants, at least for the first one.

On the basis of the characteristics described above, the pros and cons of domain modelling in technical domains (especially in LPS domains) can be stated as follows: The technical domain modelling operates on one hand, on an artificial, formal, rather limited, and relatively well controlled, but on the other, very dynamic environment. The features discussed in this chapter are:

- The domains contain a large number of domain-specific concepts and terminology that the users may be totally unaware of.
- The separation of terms from a KR model is more obvious (in these domains than typically in other domains).
- The KR model is (more or less) formally defined. Thus
- the level of specificity and semantic homogeneity of the model is good and
- the information needed for modelling is often available in electronic form.
- The mapping between model concepts and terms is more controlled.
- The mapping between model concepts and text structures is often available.

- The available mappings may have to be used for the creation of TRS before documentation is available.
- The full utilisation of the discrete model semantics requires version and configuration control.
- The rapid pace of product design, version development and automated document generation entails that modelling is automated, at least to some extent.

As stated above, some of these characteristic features, such as speed of changes or need for configuration control, are specific to a large product documentation, where as some are merely differences of decree. For example, many other scientific domains do also contain a large amount of domain-specific terminology that may be formally defined. Also many non-technical domains use databases containing some kinds of domain descriptions, such as organisation structures or product databases in commercial organisations. Thus the characteristics should not be taken as a list of features appearing exclusively in a large product documentation.

6.2 Existing Approaches

In general, there are several approaches to the construction of a domain model supporting text retrieval in a new domain. Typically, the documentation of the domain in question is used as the primary source of knowledge although some approaches focus also on expert decision making and human intuition.[52] The major approaches can be listed as follows:

- utilisation of previous domain models (e.g., thesauri),
- "top down" thesaurus construction,
- knowledge engineering approaches,
- "bottom up" thesaurus construction based on user warrant,
- "bottom up" thesaurus construction based on literary warrant,
- model extraction from text using statistical methods and clustering,
- model extraction from text using NLP, and
- combinations of the previous.

The approaches to thesaurus construction (cf. pp. 18-) included the use or incorporation of an existing thesaurus or building one anew. In cases of in-house technical domains (LPS), the incorporation of an existing thesaurus from outside the organisation does not help much, since the product- and system-specific terminology is inherently product-, system-, and thus also organisation-dependent. Within an organisation, the use of a domain model hand-crafted for a product as the basis for building a model for another product is possible by incorporating the differences in the two products or versions to the models. This

[52] This is true also in hypertext creation when it is concidered as domain modelling. Hypertext generation and definition of document structures were discussed in previous chapters.

task may still be too large or time-consuming to be performed manually because of the size of the models, the large number of the versions, the speed of version development, or due to lack of knowledgeable modeller.

The creation of a new thesaurus follows either the manual "top down" approach or either one of the alternative "bottom up" approaches. The top down approach is a fairly feasible method for small thesauri. In large domains, it entails a huge investment of effort from the subject experts that would be needed in other high priority tasks, and tends not to produce as good results as the bottom up approaches.

The problems of organising large amounts of knowledge have been addressed by various studies under the titles, such as "knowledge engineering" and "knowledge acquisition" [116, 134, 163]. In these studies, the effort needed for knowledge acquisition has been recognised as the bottle-neck in building expert systems and knowledge-based systems. The various tools, formalisms, and methodologies aim mainly at representations that would be natural for the user and at the same time computationally feasible to be used by computers. Many of the methods are fairly efficient when the size of the models is within tens or, at most, a couple of hundreds of entities, and the process is aided with sufficient tools. The problems of maintaining consistency with large systems necessitate the use of structured modelling techniques. Still, the process of acquiring all the necessary knowledge from an expert or experts is extremely elaborate, cumbersome, and expensive.

In practice, the bottom up approach to thesaurus construction cannot be based on user warrant in LPS domains, i.e., it cannot be based on users potential information requests. First, this approach shares some of the problems of the previous approach, i.e., elaborate knowledge acquisition, conflicting knowledge etc. The major problem is that the users are not familiar with the special terminology of the domain. They cannot list the terms they would like to use in requests since there is no common knowledge available about the terms the designers have just invented. They may even have difficulties in defining the kinds of terms they intend to use as they are unfamiliar with the conceptual structure of the domain (LPS). In several cases, the design organisation or the previous users may be questioned to gain information on the kinds of terms that are used. They still do not probably know the terminology used in a new design or in a new version of an old design.

The bottom up approach to thesauri construction based on literary warrant has also serious problems. It may not be possible to perform the first phase -- collection of terminology -- in due time since the documentation of a large system may not be available until the system is ready. This means that the elaborate and time-consuming creation of BT, NT, RT, and synonym relations has to be performed in zero time by the busy domain experts. In some cases the documentation is delivered document by document only after a large system or product is delivered. The problems are further complicated when the variations of the product should be made visible in the domain model. Not only terms

should be added and deleted but also relations between them. This is an elaborate task requiring knowledgeable people.

Tables of term co-occurrences generated from a document database can indicate which terms are "related" in an associative sense. This information can be used as an aid in manual creation of thesaurus relations or in an automated process, such as hierarchical clustering. Naturally, the classes formed by statistical procedures will be much less pure than those of a conventional thesaurus; a group of words that strongly co-occur may include genus/species, part/whole, and syntagmatic relationships, i.e., the generated relations are rather heterogeneous and noisy. Statistical procedures may also loose important relations between concepts that may be documented only in one text segment referring to the concepts only once -- thus not exceeding threshold values. Multi-word phrases and names are also problematic for single-word based approaches. As the aim of technical documents is to communicate as much information in as small space as possible, the important relations between concepts may be documented as a summary table, such as parts of the system and tehir sub-parts. This kind of document strucures mix term co-occurrence statistics. The correct interpretation would be to consider only the terms in the same row to co-occur. In these cases the use of formal or structural information is needed.

There are also various approaches that aim at extracting knowledge from text using NLP/NLU techniques [74, 79, 134, 144, 162].[53] These methods are typically very general, but have several serious problems that can prevent their use in practice, e.g. consistency and completeness of the model [74, 144, 162] and the amount of hand-crafted knowledge needed before the performance of the automatic extraction is on an acceptable level. projects, such as CYC [12, 129], have been working on general-purpose automatic knowledge acquisition from texts for years but have not reached sufficient level of world knowledge so far. The approaches focusing on narrower domains have succeeded somewhat better than the general ones, but have still major limitations (cf. pp. 43-).

In general, the approaches using human knowledge in domain modelling tend to be elaborate, and the only real source of interview knowledge seems to be the designers of the system who have created the conceptual structure and the terminology. The approaches extracting domain descriptions from text do not produce very accurate or selective models, especially if not all of the specific documentation configuration is available. Further, even if the terminology is extracted from the design information systems, organising this raw material is still an elaborate task. The requirement for accuracy and completeness are emphasised in LPS domains that contain vast amounts of specific terminology and concepts. Thus we can state that neither the contemporary thesaurus construction methods nor the other approaches provide a feasible method for domain modelling in technical domains that include product- or system-specific terminology and information. The major disadvantages

---

[53] This has been studied also in the SIMPR project within the context of SCES, in the form of facet creation from extracted index phrases.

are the vast amount of human effort needed and the lack of means for specifying, acquiring, and controlling an enormous number of highly specific concepts and their relations in each system and documentation configuration in hand.

6.3 Utilisation of Design Information

The artifact warrant implies that closer integration of design databases and the text retrieval domain models would offer an adequate basis for a feasible domain modelling approach in LPS domains. Figure 12 illustrates this idea by describing data flows during the design and use of a product. Instead of the domain models being constructing manually by the experts or being extracted from the documentation (dashed lines), most of the models are extracted -- more or less automatically -- from the design databases containing structured information. The models are then used in aiding the users to locate the correct information from the documentation (wide line).

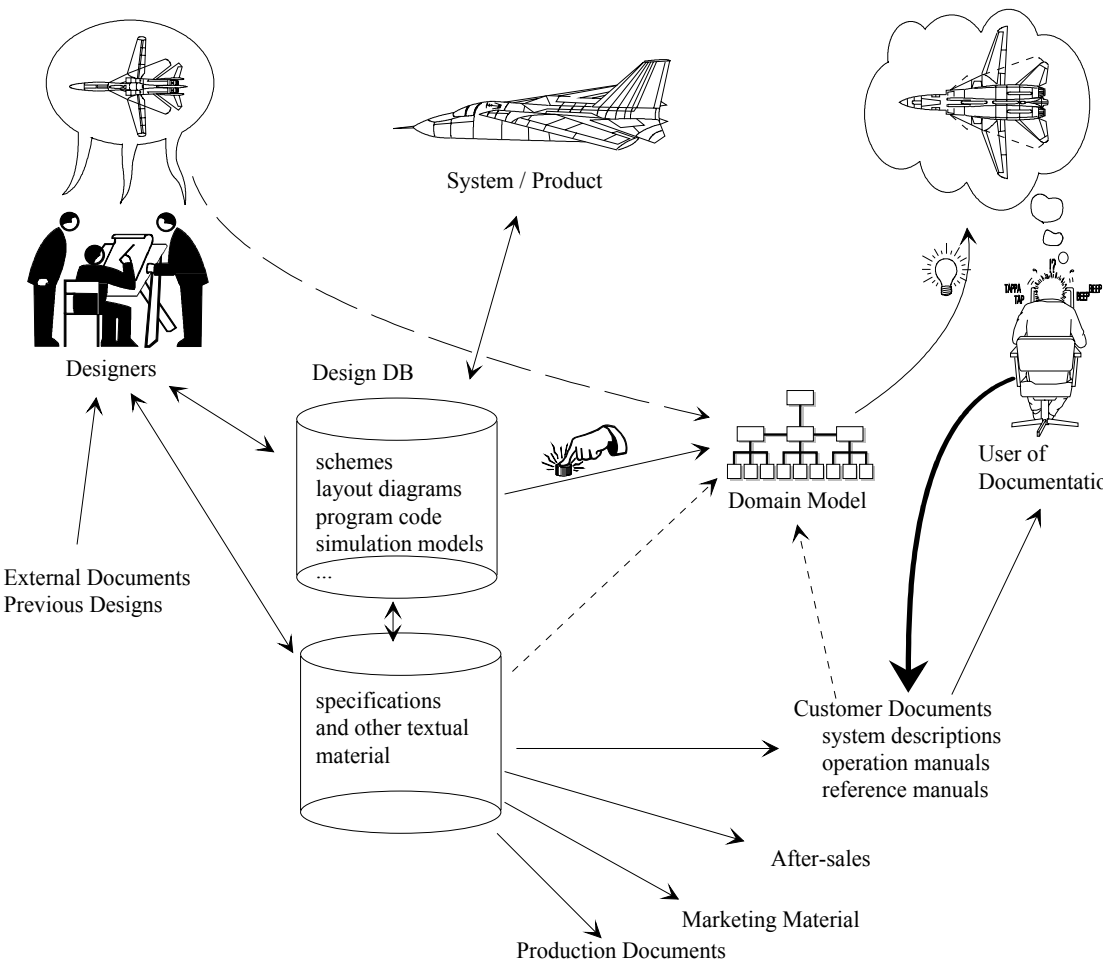

Figure 12. Use of design information in generation of domain descriptions.

There are clear advantages in the integration of domain modelling to other design activities. First of all, the information about the design concepts can be gathered semi-automatically and their relations can be used for structuring the domain models. The process

can utilise the formal semantics and completeness of the design databases. This gives basis to a kind of "closed world assumption"; the enumerated lists of concepts or relations defined in design databases can be expected to be exclusive. This is likely to require integration of the version and configuration management methods of external databases to version development of domain models. Mapping the concepts of a domain model to the terminology and possible documentation conventions is at least as important as the construction of the domain models as such. This can often utilise conventions, such as specific document structures, markers, terminology conventions and other similar means described above.

In general, the human effort needed to manipulate large volumes of rather heterogeneous information can be automated to some degree. The human intelligence needed for the model construction is reduced to the selection and integration of the information available for a specific domain. It is also most likely that parts of the domain knowledge relevant for text retrieval purposes is not available in existing information systems -- at least not with less effort than would be needed to hand-craft the same information. Thus human interaction is still essential in modelling (as well as in transferring domain knowledge) although these people have intentionally been left out from Figure 12.

The focus of the rest of this thesis is on defining the means with which to realise the scenario described above. This includes designing the means to utilise the available domain modelling information. An equally important aspect is the means to utilise information about the relation of concepts and terminology and/or documentation practices in mapping the domain models with texts. Further, because of the probable incompleteness of the available information, the methods should be flexible enough to enable partial modelling by hand.

6.4 Knowledge Representations for Domain Modelling

6.4.1 Requirements for a KR

The framework described earlier proposes some epistemological requirements for knowledge representation used in domain modelling for text retrieval purposes, mainly concerning the organisation, string-independency, and the navigation-oriented use of the representations. From the economical point of view, a domain model should be efficient to create, map to texts, and it should effectively and efficiently support text retrieval. According to the previous sub-chapter, domain modelling in technical domains should be based on the artifact principle, i.e., the use of the existing (design) information in model construction and in mapping the models with the tests. This sub-chapter summarises fifteen general requirements for KR supporting text retrieval and for closely KR-related modelling and retrieval activities. There are also five additional requirements considered to be specific

to technical documentation. The requirements for the structure and organisation of representations for a domain's conceptual knowledge are:

1. Expressive power
   Formalism should enable the representation of the central concepts of the domain, such as entities, tasks, and processes.
2. Common knowledge transfer
   The representation should give the users an overview of the conceptual structure of the domain from a common collective perspective.
3. Meta model minimisation
   The knowledge representation formalism should contain only a minimal set of structures needed by KR & TR to minimise cognitive overload.
4. Semantic homogeneity
   The representations should provide similar interpretation of similar representations by humans and computer interfaces, modelling process should ensure exclusiveness of enumeration, etc.
5. Context-based identity
   The representations should enable the identification of the concepts based on their context in the domain / world description.
6. Textual independence
   The representation should be as independent of the features of natural language text and vocabulary usage as possible.
7. Multiple hierarchies
   Representation of several overlapping hierarchies and taxonomies among the concepts should be supported.
8. Non-hierarchical relations
   Representation of various non-hierarchical relations between the concepts should be possible.
9. Navigation
   The representations should support navigating and browsing among all the relations.
10. Searching
    It should be possible to search for representations that are related to given concepts with given relations. If strings or other literal values are used, the representations should support their search.

11. Size effectiveness
    The representations should support retrieval of information from a large documentation by compact models.

12. Filtering and restructuring
    Users' cognitive overload should be minimised by providing them with views where only a subset of representations is visible, possibly in a restructured form.

13. Incrementality
    It should be possible to use the representations for effective and efficient text retrieval even if they are incomplete or are created gradually. Also changes in the representations should be possible.

14. Creation efficiency
    The representation should support efficient, preferably automated, creation and modification of models.

15. Mapping method combinations
    It should be possible to use a combination of several text representation and mapping methods (e.g., IR, NLP, and document structures) with KR.

The special features of technical (LPS) documentation retrieval and domain modelling emphasise the following properties of the KR and the combination of KR and the other parts of a system. Of these, I-II are related with text retrieval and III-VI are (partially) related with the artifact warrant in domain modelling:

I. Problem-orientation
   KR should optimise the time needed to retrieve the first relevant text.

II. Retrieval resource minimisation
    The representations should enable text retrieval in standard computer platforms, i.e., PCs.

III. Database reuse for modelling
     It should be possible to use valuable structural information of existing electronic databases in automating the creation of domain models.

IV. Database reuse for term mappings
    KR and the mapping methods used should enable the creation of mappings requiring limited human effort utilising also the mappings between design data and terminology.

V. Database reuse for structure mappings
The representations should enable efficient integration of KR with mapping methods, especially with methods utilising formal properties and structural features of the texts.

VI. Version development and configuration control
The representation should enable/provide version and configuration control functionalities similar to the functionalities of other product/system information for achieving accurate and product-specific domain models.

6.4.2 Shortcomings of Current Approaches

None of the existing approaches satisfies well enough the requirements stated above. Although each individual system has characteristics it's own, the typical problems of each major approach are listed below. The reader should also notice the dependencies of the requirements. For example, a system with limited facilities for the representation of relations cannot support advanced searching based on relations and related concepts. (For a somewhat more detailed, descriptive listing see [231].)

The subject classification systems provide typically only one hierarchical view to the domain and the faceted schemes do not support browsing non-hierarchical relations. The vector space and statistical methods lack the mechanism to transfer information about the domain's structure to the users -- knowledge representations are considered to be domain-dependent and the focus is on generic text representation level techniques. Of the information science methods, the advanced thesaurus constructions with facilities to represent several relations between concepts are closest to the requirements. A major shortcoming is that the string independence is violated by the integration of the knowledge representation and mapping level functionalities. Also, the tools for navigation and browsing tend to be rather inadequate [214], partially because of the limited set of relations, but mostly due to implementation-dependent issues.

The KR models used for NLP purposes provide rich representation facilities that can efficiently be used for the representation of the central concepts of texts as well as the relations between the concepts. Typically these representations are oriented towards full text understanding and thus organised towards efficient description of linguistic knowledge and categorisations rather than efficient representation of concepts central to text retrieval purposes. In practice, this means that the representations are fairly elaborated and most of them support only the search paradigm, i.e., the comparison of representations derived from texts with the representations created for a query. This requires significant computational resources during the text retrieval process. Theese approaches do not efficiently support browsing and navigation thus making their use for text retrieval purposes inadequate. In addition, the NLP oriented representations cannot easily be adapted in environments where a variety of mapping and text representation methods are used (including use of document

structures) to balance the great processing power requirements of NLP methodologies by selective processing. Thus, in general, a separate, text retrieval-oriented representation of central concepts would be more efficient, even if it were used in combination with (as in [12]) or as a part of a KR used for NLP.

The analysis of hypertext approaches showed that they combine the knowledge and text representation levels. This improves the retrieval efficiency in small amounts of text but degrades soon as the volume of texts grows. Thus the explicit separation of texts and the concepts is necessary. The document structure approaches suffered from problems similar to small pre-defined classification systems.

From the domain model construction perspective, the major problem of most of the KR models is the effort needed to create the domain representations. The general purpose representations have not adopted model construction methods operating in accordance with the artifact warrant principle, although this would -- at least to some degree -- be possible.

7 D&T MODELS -- A KNOWLEDGE REPRESENTATION SUPPORTING TEXT RETRIEVAL

7.1 Approach

This and the following two chapters evaluate the requirements for technical domain modelling process and knowledge representations for text retrieval systems by implementing and evaluating a KR-level representation for domain models and a modelling approach in accordance with the artifact warrant principle. The domain modelling system described in this chapter has been designed and implemented jointly by our research group at NRC, within the scope of the project SIMPR. The further analysis and methodology development can be considered as the authors work, although they have been partially implemented by other members of the group and reported jointly as SIMPR reports.

The following model aims at satisfying the requirements stated above. The selections made along the design process aim at:

- minimising the text retrieval effort by emphasising the preparations in form of proper creation of a KR model,
- minimising the effort needed for the modelling by utilisation of existing electronic databases in accordance with the artifact warrant principle,
- optimising the level of elaboration of the model by using a compact model that is efficient to build in a combination with less user-friendly text representation level techniques that cover the potential lacks of the model during text retrieval.

We also chose to:

- support user-driven navigation, browsing, and searching within a KR model,
- use models with no deductive capabilities or feed-back features,
- separate between a definitional and a declarative component of a model (i.e., scheme and data),
- use attribute values and relations between concepts to describe the concepts,
- represent concepts and their relations as objects,
- support instantiation and generalisation (subclass relations) directly,
- provide an open-ended facility for representation of other relations including other taxonomies, such as aggregation.

Within the previous selections the modelling system can be used within various system architectures and purposes. With respect to the context where the modelling system is to be tested, we considered the alternative mappings between the KR model and text representatives and text. The following decisions were taken:

- require processing during creation of mappings between a model and the texts and require no processing capabilities (such as NLP) during retrieval, (and thus:)
- enable use of separated processing and retrieval environments,
- integrate and store the mappings with and as a part of the models,
- support two way mapping creation, i.e., both from the texts to the models and from the models to text representations and texts.

7.2 Domain and Task Models

Based on the requirements and design selections described above we designed and implemented a system for building domain models for text retrieval purposes. The system is called *domain and task modelling* (DTM, D&T modelling) system and the models built using the system are called *domain and task models* (D&T models) or simply domain models or models. These models are a data model of the information that we consider essential for text retrieval in technical domains or can be represented better by formal KR than by text. The rest of this chapter describes the D&T models, ways to use them, and their properties.

A domain and task model is a description of the central concepts of a domain. It describes the organisation of interrelated concepts and provide users with documentation describing each of them. During retrieval the users browse a model in order to find the concepts they are interested in. This is needed especially in technical domains, where the users of documentation are often unaware of the detailed structure of a domain and especially unaware of the concepts, terminology, proper names, and naming conventions used in specific documentation.

The models consist of three parts: 1) a definition scheme, 2) an instance world, and 3) a view type hierarchy. The definition scheme defines the existing concept classes of a domain, attributes characterising each class of concepts, and defines relations between the instances of concept classes. The instance world contains concept instances characterised by attribute values and relations between the concept instances. The instance world is a model of the domain, its entities, their properties, and their organisation. The view hierarchy contains view types, that are used as labels of relations. These labels limit the amount of information displayed for the user by selecting subsets of the relations to be displayed in a view, for example, in views used for diagnostic or installation tasks. Figure 13 displays the overall structure of the models. The definition scheme is displayed on the top and the instance world in the middle. The view hierarchy is not included in this diagram. Instead,

there is a layer describing the documentation as a set of interconnected text units that are referred to from the model.

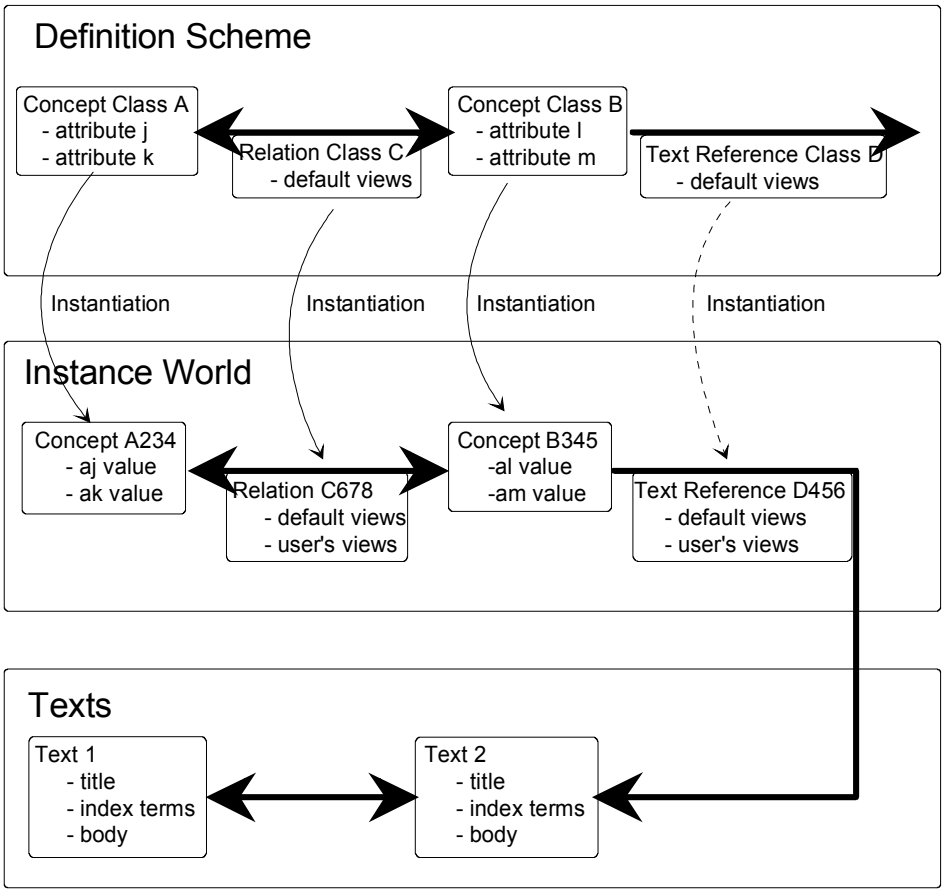

Figure 13. A descriptive diagram of the D&T models.

7.2.1 Definition Scheme

As a data model the definition scheme is close to an extended entity-relationship (EER) scheme or object diagrams in Object Modelling Technique (OMT) notation [182]. It defines the concept classes of the domain and relation classes between them. These correspond to entities and relations in an EER model or classes (excluding operations) and associations in a class diagram of an object diagram in OMT. The structure of a concept class is defined by its attributes. They provide default values for the concept's instance's attribute values. The concept classes form a tree or a directed lattice which defines an inheritance hierarchy for the concept classes. Attributes of a concept class and their default values are inherited by its subclasses.

Relation classes are defined between two concept classes. They specify the relations which must or may exist between two instances of those concept classes to which the relation classes are attached. For example, the relation class C in Figure 13 can define that there must exist at least one relation (like relation C678) from each instance of concept class A and its subclasses (like concept instance A234) to some instance of concept class B or its

subclasses (like concept instance B345). Also the cardinality of concept instances' relations can be restricted by the relation class. Like the concept classes the relation classes are created for each domain according to the preferences of the domain experts. The relation classes can be based on any kind of causal or ontological relation that is valid in the domain.

Text reference classes are stereotyped methods to access text database by any of the instances of a concept class. Each instance of a concept class is attached to a text reference based on the text reference classes attached to its concept class. The text reference classes are described in detail later on.

7.2.2 Instance World

The instance world consists of concept instances, relations, and text references. Concept instances represent central entities of the domain, such as individual products and their parts. Every instance in the instance world is an instantiation of one of the concept classes specified in the definition scheme, and has the attributes described by the corresponding class.

The relations are instantiations of the relation classes defined at the definition level. The relations may be created only between instances of the concept classes (and their subclasses) that the relation class was defined for. Multivalued relations are an abstraction of a set of binary relations. The number of relations starting from and ending at each concept instance must satisfy the constraints imposed by the instantiated relation class. The relations are typed by the type hierarchy described below.

From each concept instance, there can exist several references to texts. Texts are referred to by the text references that are either explicit references to a named text or computational references defined by the procedure contained in a text reference. The result of this computation is an identifier of a text in a form that can be accepted by the storage system. From the model point of view the texts are manipulated as (a network of interconnected) objects with unique identities.

7.2.3 View Types and Type Hierarchy

The number of relations between concept instances as well as the number of text references from a concept instance to texts can easily grow to large quantities that are impractical to display and manipulate interactively through a graphical user interface. View types are a mechanism to reduce the cognitive overload by restricting the displayed information to relations and text references included in a view. Thus only the information relevant for the current retrieval task is accessible for the users, who browse or navigate the model. Each relation and text reference is included in the views that the corresponding class (relation class and text reference class, respectively) is in. In addition, the individual relations and

text references may be included in extra views. The information about being included in a view is presented as a property of each of the relations and text references.[54]

7.2.4 Text retrieval using D&T Models

Domain and task models are used to access texts using structural information in a hypertext fashion, in graphical browsing and retrieval. The text retrieval with D&T models can be divided into two main phases. First, the user locates the interesting concept and secondly, retrieves documentation related to it. The first phase starts from the definition scheme, where the user selects an interesting concept class. After selecting an instance of the class the user is able to browse relations between the concept instances in the instance world and to navigate a path to an interesting topic represented by a concept instance. In the second phase of the process the user retrieves documentation related to the selected concept instance. These links are implemented by the abstract text reference objects.

During the text retrieval process most of the time is spent on the model browsing and only a small portion is used for the retrieval of the documents related to a concept instance. The paths used in the navigation vary according to the users and cannot be created in advance. Instead, linkage between texts and concept instances can be carried out and the links can be stored in an efficiently accessible form before the actual retrieval process takes place.

7.2.5 An Example

Figure 14 represents an example domain model for a car domain. The documentation in a database includes texts describing different car models, their parts, maintenance, etc. In the top part of the Figure the definition scheme contains concept classes, such as *car model*, *engine*, *gasoline*, and *body*. These concept classes are supposed to be the central concepts of the domain based on the opinion of the domain experts and terminologists. There exists also relation classes, such as *part-of* and *uses*. They define that for each *car model* there exists at least one *engine* that is used (*part-of*) in that model, that each *engine uses* some *gasoline*, etc. There exists also a text reference class that defines an access method for documentation describing any engine.

Let us now suppose that we would like to read documentation about the motor of our car. With full-text search we would perhaps not be able to locate any texts where the term "motor" exists. After consulting a thesaurus the search with term "engine" retrieves lots of

---

[54] The view type hierarchy could also be represented as a relation class hierarchy, analogously to the concept class hierarchy. Unlike the concept classes, the relations have a uniform structure, that does not gain advantage from inheritance. Instead, use of a relation class hierarchy would cause some overhead, like require creation of new relation classes for each combination of view types used in a model.

texts. A combination of the model name "Saab Cabriolet" and the term "engine" does not produce any hits.[55] What to do next?

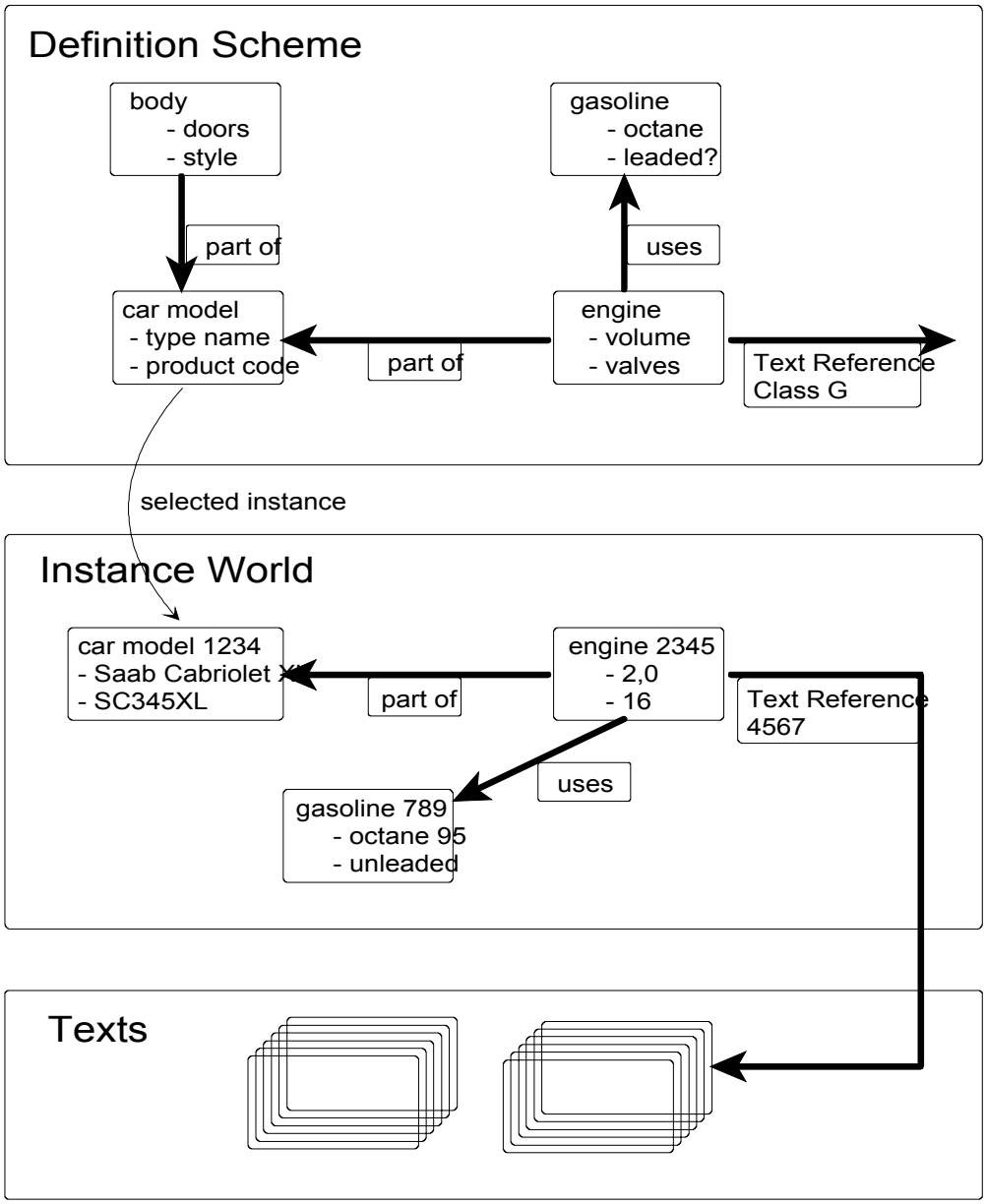

Figure 14. Retrieval of car manuals using D&T models.

We continue with the domain model. First we consult the definition scheme, where we are able to find the interesting concept classes, *engine* and *car model* based on their names, relations to other concept classes, and their attributes.[56] As we do not know which of the 100 different *engine* alternatives (instances) we precisely do have in our car, we start with

[55] In our fictive example the reason for this is that the manuals do not refer to the car models when discussing the engines. And when they do, the reference is by product code, not by type name.

[56] The concept class *engine* was found based on its' *part of* relation to the *car model* and based on attributes *volume* and *valves* even when we were trying to find a concept class like *motor*.

the information we do know, the attributes of the *car models*. By displaying a list of *type names* (attribute) of *car models* containing "Saab" and "Cabriolet" we are able to find our model among the four alternative Cabriolet versions.

We continue at the instance world with the instance of *car model* concept whose *type name* is "Saab Cabriolet XL". Next we display the *part-of* relations starting from it and find the information represented in the middle of Figure 14. Among the instances displayed we find three engine alternatives, of which we are able to choose our engine based on the *volume* (attribute) of the motor.

From the engine we access the texts using the text reference defined by the text reference class attached to the concept class *engine*. In this case the system retrieves fifty documents which we wish not to go through before finding the one needed. Thus we limit the text references (and relations) displayed to those included in view *maintenance* (selected from a view type hierarchy that is not displayed in Figure 14). From the five remaining text references we are able to locate the information we need.

If we would like to know the kinds of fuel that can be used in the motor we would not have to consult the documentation. In this case it would be sufficient to display *uses* relations from our *engine* to the instances of the *gasoline* concept class. Thus the model can be used also for learning the domain, although the primary aim is to aid in text retrieval.

In this example we followed the *part-of* relations, that form a hierarchy from a product (the car model) to its parts, parts of parts, parts of parts of parts, etc. There might also exist alternative, overlapping hierarchical relations based on functional categorisation, functional organisation, and material of the parts, as well as other taxonomies.

We also examined non-hierarchical relations from the motor to the gasoline it uses. The model might as well define the functional dependencies of the parts, data flows in a micro chip controlling the injection, and other relations based on some important ontological feature of the domain.

From this example we can notice that the identity of the concepts and concept classes is not determined by their names, but also on their attributes, attribute values and relations to other concepts. Thus the user is able to navigate to the concept class *gasoline* based on its' relation to *engine* and attributes *octane* and *leaded?* even if the term fuel would have been more familiar to the user. Also the text references from concept (whose *type name* attribute is) "Saab Cabriolet XL" may utilise other attributes of the concept as well as relations to other concepts.

The vocabulary independent identity of concepts can be utilised in multi-lingual applications. Each of the languages needs only separate versions of the textual attributes -- the numeric attributes and relations remain the same. For this reason the translation of textual attributes in D&T models is not as critical as translation process in approaches where unique names alone determine the identity of a concept. As far as the concepts are language / culture independent, as they typically are in technical domains. If the same documentation

is translated to several languages the text references created between the model concepts and the texts written in one language can be used also for the corresponding text segments in other languages, i.e., the text segments translated from the primary language text segment referred to by the concept. They describe the same concept and occupy the same position in the translated documents as the original text segment. This can save lots of human work and computer resources needed for the mapping process as well as effort needed for tailoring the methods for several languages. Further, the language used for the primary mapping creation can be chosen so that the creation of the primary mappings is as effective and efficient as possible.

7.2.6 Formal Description of D&T Models

The structure of the objects in a model is dependent on other objects of the model due to inheritance and instantiation. If we neglect this kind of dependencies, the description below gives a good overview of the models in some fixed state. The formalism used is BNF-like (for further details and implementation issues see [91]):

```
<concept-class>          ::=   '(' CONCEPT -CLASS
                               <concept-name>
                               <super-concepts>
                               <attributes>
                               <relation-classes>
                               <text-reference-classes>
                               <text-references> ')'

<super-concepts>         ::=   '(' <concept-name>* ')'

<text-reference-classes> ::=   '(' <text-reference-class>* ')'

<text-references>        ::=   '(' <text-reference>* ')'

<attributes>             ::=   '(' <attribute>* ')'

<attribute>              ::=   '(' <attribute-name>
                               <default-value>
                                <data-type> ')'

<relation-classes>       ::=   '(' <relation-name>* ')'

<relation-class>         ::=   '(' RELATION-CLASS
                               <relation-name>
                               <concept-name>          /* from concept */
                               <concept-name>          /* to concept */
                               <concept-cardinality>   /* from concept instances */
                               <concept-cardinality>   /* to concept instances */
                               <view-types> ')'

<concept-cardinality>    ::=   '(' <integer> <integer> ')'   /* min & max */

<view-types>             ::=   '(' <view-name>* ')'

<view-type>              ::=   '(' VIEW-TYPE
                               <view-name> <super-view-types> ')'
```

```
<super-view-types>      ::=     '(' <view-name>* ')'

<concept-instance>      ::=     '(' CONCEPT-INSTANCE
                                <concept-name>
                                <instance-id>
                                <attribute-fillers>
                                <relations>
                                <text-references> ')'

<attribute-fillers>     ::=     '(' <attribute-filler>* ')'

<attribute-filler>      ::=     <attribute-name> <attribute-value>

<relations>             ::=     '(' <relation-id>* ')'

<relation>              ::=     '(' RELATION
                                <relation-class-name>
                                <relation-id>
                                <concept-instance-id>     /*from concept instance */
                                <concept-instance-id>     /*to concept instance */
                                <view-types> ')'

<text-reference-class>  ::=     '(' TEXT-REFERENCE-CLASS
                                <concept-name>
                                <text-definition>
                                <view-types> ')'

<text-reference>        ::=     '(' TEXT-REFERENCE
                                ( <concept-name> | <instance-id> )
                                ( <text-definition> | <text-reference-class> )
                                <view-types> ')'

<text-definition>       ::=     '(' REFERRED-TEXTS
                                ( <text-id> | <mapping-procedure> ) ')'

<mapping-procedure>     ::=     <keyword-query>
                                | <key-phrase-query>
                                | <document-structure-navigation>
                                | <lisp-clause>
```

7.3 Uses and Users of D&T Models

7.3.1 Position in the Framework

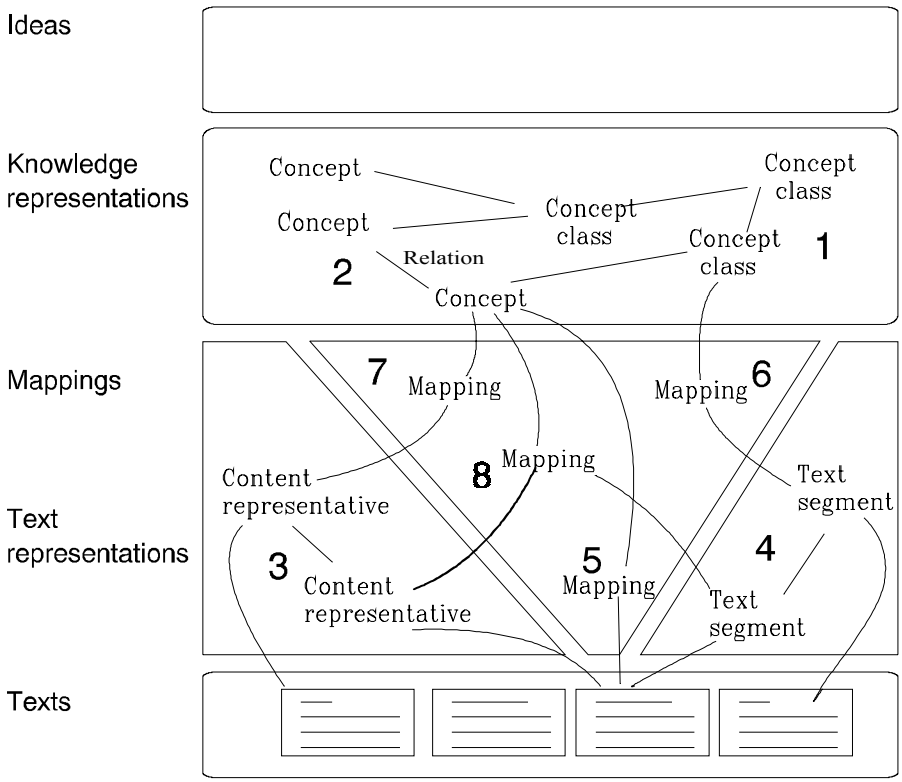

Figure 15. D&T modelling in the framework.

Figure 15. represents the D&T modelling system from the point of view of our framework for text retrieval systems presented earlier. The models are purely representations used at the knowledge representation level, including the definition scheme as the declarative component (1) and the instance world as the component describing the major portion of the conceptual knowledge (2). In addition, the text references included in the model can be classified as the major mapping components used during retrieval. This means that the knowledge of best available mapping methods is stored in the model with the concepts in form of text references. The possibilities for mapping methods to be used include direct mappings to texts (5) created by a librarian, a subject classification system, or a NLU system. Dynamic mappings can be implemented as queries (7, 8) utilising index terms, probabilistic or weighted content vectors, index phrases, semantic descriptors, or other levels of content representatives (3), mappings (6) between document structures (4) and concepts, as well as mappings utilising combinations of methods (8), e.g., NLP of certain text fields.

7.3.2 Purposes

The motivation of the D&T models is to help with text retrieval and to transfer parts of the domain knowledge, as stated earlier. This means that the models are created separately, a mapping between texts and a model is created, and the texts are retrieved using the models. In addition to this, the models can, to some degree, be used for development of two related methods, namely to support knowledge extraction from text and for text generation.

In knowledge extraction the role of the models can be defined as the framework, to which the new conceptual knowledge is added. The simplest way to realise this is to find new relations between existing concepts by locating co-occurrences of the concepts and use NLP to recognise the relations. Further, new concepts can be extracted based on occurrences of unknown terms, expressions, and phrases in positions, where typically exists only instances of a specific concept class. Especially exclusive lists of concepts related to a concept of another class are easy to recognise, that enables verification of the contents of texts with a model. For example, a list of motor types available for a certain car is easy to compare with the set of motor concepts related to the specific car concept. Thus the constraints imposed by the definition scheme are valuable for validation of these two methods.

The extraction of definitional information to the definition scheme is a more demanding task, as described in previous chapters. One major problem is that the mapping between natural language expressions and (possibly transitive) relations used in a model may utilise different levels of abstraction. For example, "a red car" in our example model is represented as a *car* instance whose *part* is a *body* instance whose *colour* is "red". Further, lots of the structure of the definition scheme can be defined in several alternative ways and the human judgement are also used for deciding whether some aspect of the domain is included in the model or not (i.e., is it useful for text retrieval purposes).

The D&T models can also be used for text generation purposes. Most of the contents of a model are likely to be relevant for any user who would read the text generated from a model. When only a limited amount of text should be generated, the view mechanism can be applied for selection of the parts of the model used for generation. An abstraction mechanism can also be constructed based on concept recognition and text generation; the information contained in a text is converted to a model and returned to another piece of text. This functionality would, though, require information which of the relations and attributes were referred in the texts, not only information about references to concepts, as is used for text retrieval.

7.3.3 User Roles

When the D&T models are used for text retrieval purposes there exists three user roles according to the main tasks: the creation of the models (i.e., preparations), the creation of mappings between models and texts (i.e., processing), and retrieval of a documents using

the models. The roles are a model editor, a text editor, and an end user. In practice the roles may overlap and a single user may perform several of the roles.

The *model editor* is a person responsible for building D&T models describing the domain. The goal of the model editor is to maintain a consistent model and to use design information in the models. The model editor uses an interactive interface for modifying any part of the model as necessary. In the definition scheme, the model editor can add and delete concepts as well as modify their attribute values and relations. This interactive development is centered around graphical browsers which display the models' concept class and view hierarchies and form editors used for attribute value editing.

The *text editor* is the person who is responsible for adding texts into the system. The goal of the text editor is to link the texts with the models. The graphical model browsing interface is relatively identical to the interactive model editing interface, aside from a more limited functionality. Graphical browsing of the hierarchies and graphs is enabled, but the ability to modify the definition level is disabled, and the modification of instances is restricted to the addition of text-relations.

The *end users* (or retrieval users) use the models mainly to retrieve documents. The tools may also be used to learn the structure of a domain by browsing the models.

8 DTM -- SEMI-AUTOMATED DOMAIN MODELLING FOR TECHNICAL DOMAINS

A D&T model which will be useful for large documentations is remarkably large even in limited domains. This means that the domain modelling problems discussed earlier (beginning from p. 73) strongly suggest the use of methods specialised for technical domains, i.e., to follow the artifact warrant principle. The two major problems discussed later two subchapters are:

- the high effort needed for construction of large models, and
- the high effort needed for linking a model with the documentation .

Before addressing these two problems the first sub-chapter estimates the size of the D&T models and compares the use of the models with traditional hypertext approach. The second sub-chapter presents the CIOS architecture for text retrieval user interfaces adopted based on the analysis. The following two sub-chapters present the methods for automating the creation of D&T models and for automating the mapping of model concepts with the relevant texts based on the artifact warrant principle. The last sub-chapter presents guidelines for their application following the artifact warrant principle.

8.1 Size of D&T Models

The SIMPR project focuses on management of large text databases. In this context the term large is interpreted as something like 300 MBytes of free text. This implies that the database will contain about 600 000 texts (i.e., texts between two headings) on average two per page. A useful D&T model for a text database of this magnitude is here estimated to have roughly:

- 100 concept classes in the definition scheme,
- 100 view types in the type hierarchy, and
- $2 * 10^4$ concept instances in the instance world.

In the number of objects this means that the size of the definition scheme is approximately two orders of magnitude smaller than the size of the instance world, which is one to two orders of magnitude smaller than the text database. This gives a reasonable way to manage the text retrieval by three levels of hierarchy and manageable amount of objects on each level as well as a manageable amount of connections between the levels. (See Figure 16.)

Each of the concept classes is instantiated on an average 200 times although some classes have also several thousands of instances and some have fewer. Each concept instance is related with about 20 other instances. Each of the instances refers to about 90

texts via text references grouped by the view types.[57] Thus each of the texts is associated on average with 3 concept instances, and the total number of entities (including number of text references and texts) is about $2{,}8 * 10^6$. These figures are partially based on our experiments with real examples described later on.

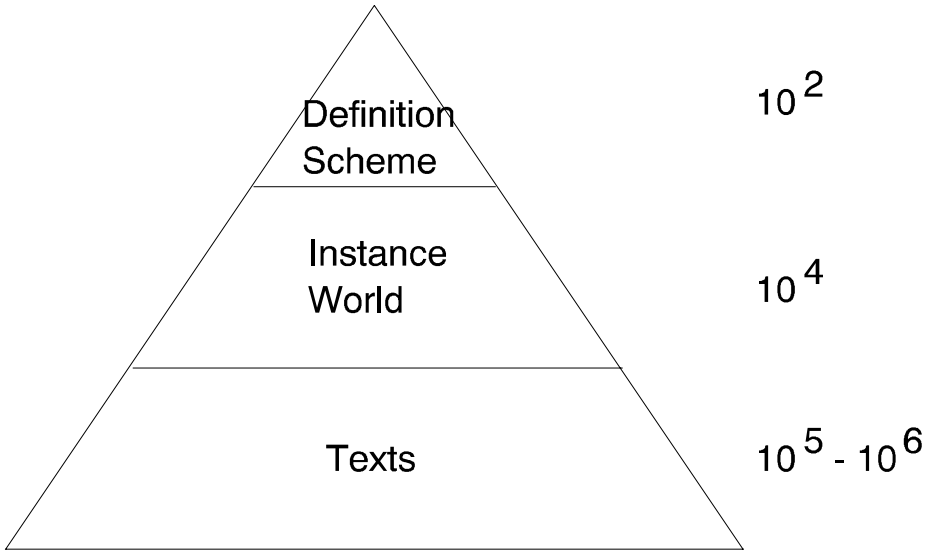

Figure 16. Orders of magnitude in D&T models.

A summary of relations in an typical example would be something like following:

- $2 * 10^4$ instantiations,
  i.e., $2 * 10^2$ instances per concept class, (1 per instance),
- $4 * 10^5$ relations between instances,
  i.e., 20 per instance.
- $1{,}8 * 10^6$ text references from instances to texts,
  i.e., 90 per instance[58] or 3 per text.

The implementation of traditional hypertext links between texts would be simple based on the relations and text references of the model. The hypertext links are now present implicitly, in the form of a) texts that are referred to by the same concept instance and in the form of b) texts referred to by related instances. The creation of explicit relations between texts would produce about $8 * 10^7$ or $1{,}6 * 10^9$ links between texts when using strategy a) or b), respectively,[59] i.e., the number of hypertext links would be 28 or 570 times the

[57] Also the concept classes may have text references, but they are ignored in this calculation for simplicity.

[58] 90 texts per concept instance (on an average) is somewhat smaller figure than the typical number of texts considered to be relevant for a concept in a typical TRS [185]. This number is still smaller, if only a portion of the total documentation is searched.

[59] In alternative a) this means creating links from each of the 90 texts referred by an instance to each of the 89 other texts for each of the $2 * 10^4$ instances / 2. In alternative b) creating links from each of the 90 texts referred by an instance to each of the 90 texts referred by each of the 20

number of all other entities in the system!! Although these figures are only rough estimates and the storage of links between texts is not very expensive, they still indicate that the use of a separate modelling component has some clear benefits.

Text retrieval based on model browsing is also influenced by these levels. The process guides the user through these three levels from smallest to largest. At each level the user is able to browse hierarchical and non-hierarchical relations.[60] In general, the domain model is oriented towards an object-oriented database that can be inspected from several points of view depending on the information need of the user.

8.2 CIOS Text retrieval Architecture

The D&T models represent an object-oriented approach for domain modelling. This approach can also be expanded to text storage. Although this is not essential for the use of D&T models, it displays some benefits from the user interface point of view. In this approach the texts are represented as a network (essentially a tree) of text objects. Each of them is linked with its sub-chapters and has attributes, such as heading, text string, and keywords. The references from models and index files point to these text objects that can be displayed and manipulated using graphical user interfaces independently from the text strings. The text contents of the text objects are stored as parts of the documents in their original format. Each text object is able to access its contents as a separate text segment or as a part of the document, in the original context. This approach enables users to browse and manipulate the text objects fast before accessing the contents of selected texts. It also enables storing the texts in original format under a variety of storage systems. This is analogous to hypertext systems, where separate browsers are used for displaying the headings of text nodes. By these means we end up in an architecture, where the users see four hierarchical levels of entities: the concept classes in the definition scheme, the concept instances in the instance world, the text objects, and the text segments. This CIOS (classes-instances-objects-segments) architecture is represented in Figure 17.

In this architecture the two topmost layers correspond the knowledge representation level in the framework for text retrieval representations. The text objects are document structure representations of the text representation level and the text segments correspond to the raw text at the text level. The mappings between the KR and text representations, i.e., between concept instances and text objects, are not visible for the user retrieving texts. If a user wants to search for texts containing some specific terms, (s)he starts with a query string and ends up with a set of text objects.

---

related instances for each of the $2 * 10^4$ instances / 2. These figures include some overlapping.

[60] A class hierarchy at the definition level, hierarchical and non-hierarchical relations between parts of the model on the description level, and a heading hierarchy on the text level.

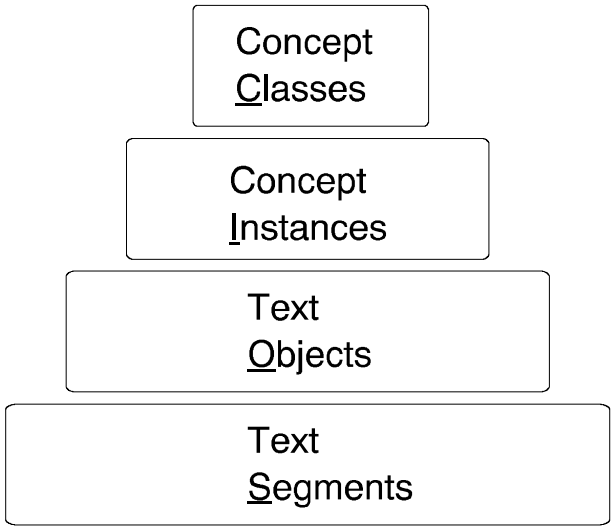

Figure 17. The CIOS architecture for text retrieval.

From the users point of view all the three topmost levels of the CIOS architecture contain hierarchies of objects that can be browsed or navigated through similar interfaces. The two topmost levels contain also non-hierarchical links between objects, whereas the text objects are stored as simple hierarchies. Implementation of non-hierarchical links between the text objects would mean implementing hypertext links between the texts. The disadvantage of the hypertext links would be the increase of number of connections between objects by an order of magnitude (cf. p. 99).

8.3 Creation of D&T Models

8.3.1 Introduction

The domain modelling analysis for technical domains (cf. pp. 73-) suggested use of external databases as the source of domain-specific concepts/terms and their interrelations. The feasibility of this approach is evaluated by implementing tools supporting semi-automated use of design database information in building DTM domain models. The domain and task modelling system uses a combination of manual creation of the models with the utilisation of electronical information sources. The leading principle in the modelling system is to limit the manual work to the minimum and to utilise tools of the D&T modelling system to input vast majority of the data from external sources. Thus the majority of the manual work, the creation of the "real model" at the description level, is automated based on utilisation of existing structured information banks. This is preferred to the extraction from texts due to the speed of the process as well as to the easier management of consistency and completeness of the database data in comparison with the data extracted from the texts.

8.3.2 Definition Scheme vs. Instance World

One of the leading principles in the modelling system is to separate the description level and the data level to the definition scheme and to the instance world. As described earlier, the structure of the definition scheme is close to the extended entity relationship (EER) model and an object diagrams in Object Modelling Technique (OMT). This similarity enables easy creation of mappings from database schemes to the model as ER descriptions of external databases are typically available. The mappings between the database schemes and the definition scheme of a model are then used by the conversion routines to convert selected information from database tables to the instance world.

The possibility to import parts of the definition scheme descriptions from the schemes of database systems was also considered; In relational databases they are available in form of system tables. There are no principal obstacles to do this. The reasons not to do this are based on the content of the scheme. In some experimental cases we found out that the view to the database in the database scheme is not necessarily adequate for the user. Lots of the data contained in databases is irrelevant for document retrieval purposes from almost any point of view or is represented on an unsuitable level of abstraction. Thus inclusion of a database scheme as such in a definition scheme of a D&T model would bring lots of noise with the relevant concept classes, attributes, and relation classes. The preferable way is to manually go through the DB scheme and pick out the relevant concepts, their relevant attributes, and the relevant relations between them.

8.3.3 Importing Design Information

The data interchange process is implemented as two separate processes that communicate via intermediate files; the database system outputs the selected information to files that are read into the D&T modelling system. This technique is selected on one hand to enable use of the DTM system in environments, where not any direct connection between the database system and the DTM system are available due to the lack of network connections or due to the complexity of the data extraction process, for example. Now the information can be extracted separately and delivered even by cartridge tape. On the other hand there is no need for close co-operation of the systems as long as the operation is performed only once.

The incorporation of only selected parts from the DB to a D&T model has some influence on the conversion routines. As the database system produces some intermediate files, that are later read into a D&T model, it can also make required datatype conversions for the attribute values, create relations based on join operations, as well as utilise view mechanism of the DB system to produce suitable intermediate files. The intermediate files may also be filtered, reformatted, and edited manually, if necessary. Figure 18 represents the process.

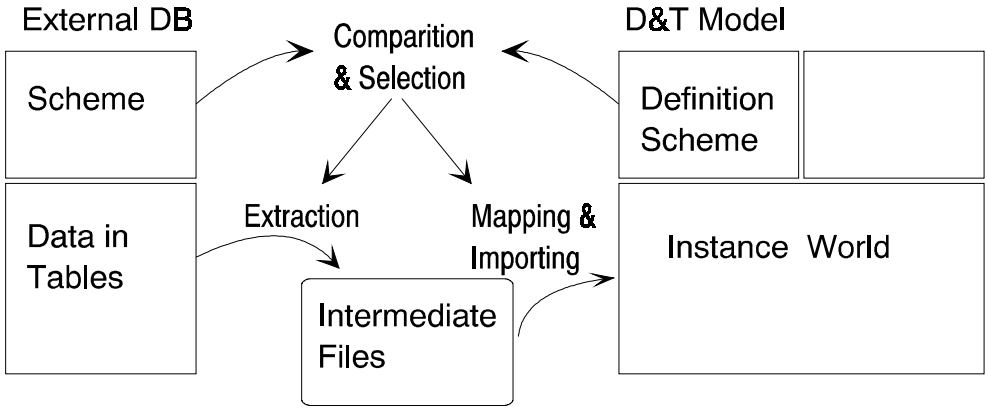

Figure 18. Model information input from a relational database.

Typically the data of a D&T model is collected from several sources, which incorporates version management problems and other consistency problems to the process. The schemes of many external databases may be conflicting and the result of merging parts of two databases in a D&T model may include duplicate information, conflicts, and gaps. These problems can be tackled partially before importing the data to the model or the constraints and datatype definitions of the model may be enforced. During the input operation the data cannot be fully validated since the importing process is a sequential operation; obligatory relations cannot be enforced during the input of an instance since the related instances may be missing for a while and the relations cannot be established before both of the instances are present.

In an optimal case the source of information is a relational database, from which the data can be extracted with SQL queries to the intermediate file. Although the relational model is commonly used in modern environments, there exist also many databases based on hierarchical and network data models. The method used in D&T model input sets no limits to the data model used, as long as we are able to produce intermediate files in a suitable format.

The use of external information sources may also utilise other storage methods than databases in the traditional meaning of the term in context of database management systems. Some organisations are using CASE tools, SDL or VHDL formats, semi-structured files, specialised file formats, or even directory structures to store structured information. Files may contain SGML tagged fields, header information that can be extracted automatically, or even program code that can be parsed in order to extract information relevant to the D&T models. The purpose of the model input interface is to offer a simple and very general interface that accepts a wide variation of data sources.

8.3.4 Operation

The person building the model, the model builder or the model editor (ME), first builds a definition scheme. For each concept the ME defines a mapping from rows of an intermediate file to the attributes of a concept. The interface program creates an instance of

the concept class for each row of the file based on the attribute mappings and default values. In some cases there may already be a similar concept instance in the instance world. In these cases the user is able to override the previous values or to use the old instance. In order to recognise the equality of instances, the model editor defines a set of attributes that define the identity of the concept instances during the conversion process. This method is similar to the definition of key attributes in database systems. After the conversion process the identity of the concepts is no longer based on the key values, i.e., two instances may have similar attribute values in the selected attributes. As an alternative to the mapping against a concept class the model editor may choose to map an intermediate file against a relation class. For this purpose the fields of the intermediate file are mapped against the key attributes of the two concept classes related by the relation class. The process produces relations between the instances defined by these key attributes analogously to the production of concept instances. The types of the created relations are defined by the default types of the relation class.

The conversion programs are started from the definition scheme, via the graphical user interface. A screen dump of the interface is displayed in appendix A. In the figure the process is in the phase, where the user is defining the input routines for importing concept instances for a concept class. The form window on the left displays the structure of a concept class. The form window on the right is used for describing the order of the attribute values in the intermediate file. The second figure in appendix A displays the situation after the intermediate file is read in. The information window has listed the attribute values read from the file and printed the message "38 instances retracted" in the end.

The utilisation of external data sources requires some additional effort in the form of data extraction as well as in definition of the mappings. If the models were hand-crafted, the same information would be needed to aid the manual building of the models. Thus the definition of mappings from the data to the models is not actually extra work. If we suppose the size of the definition scheme to be about two orders of magnitude smaller than the size of the instance world, the total effort needed for model building is likely to reduce to less than one tenth of the original, i.e., by one to two orders of magnitude! Thus the use of design information interface seems to be worth while. It may even be the only feasible method available in large cases where accurate models should be produced with limited human resources.

8.4 Creation of Text References

8.4.1 Direction of Creation

The DTM system enables creation of mappings between model objects and the texts by several means, both from texts to the model as well as from the model objects to the texts.[61] There seems not to be a single method, that would be superior to the others in all

[61] See chapter "Guide-lines for Applying KR to TR", starting from page 112.

circumstances, when quality of the result, effort needed by a human, as well as computation power required from the system are considered. Thus the approach taken in the DTM system is to enable utilisation of a variety of methods in the creation phase and to hide the techniques used under a uniform abstraction in retrieval phase.

With respect to the direction of the creation process the preferred method was to process all the texts at first and then create the mappings from the model to the texts based on the text representations. This enables use of various processing and mapping methods as well as relatively easy implementation of changes to the model with no exhaustive reprocessing of the texts. The main problem is, how to create the huge amount of these mappings even with efficient tools. The focus is on two questions: first, how to create references from a model object to the relevant texts at all and second, how to automate their creation in order to minimise the required hand-crafting.

8.4.2 Dynamic and Fixed Text References

A text reference is an abstract object of the DTM system that links a concept instance (or a concept class[62]) to a set of text objects. A concept instance may have several text references labelled by a set of view types. A text reference is either a fixed (static) or a dynamic (computational) text reference. A fixed reference contains a set of direct references (pointers, surrogates) to texts. A dynamic reference contains a definition of the method for finding the texts. This method may utilise structural properties of the texts, content representatives, statistical methods, or whatsoever methods are best suitable for the case in hand. During the retrieval this definition is evaluated to a set of direct references to texts. This set, i.e., extension of a dynamic reference, depends on the context where it was evaluated; direct references are created only to the portion of documentation that is selected for retrieval.

In the retrieval phase dynamic and fixed references look alike. The user sees only the direct references to the texts, i.e., which texts are linked to the concept instance via a text reference. Thus a dynamic text reference appears as a set of direct references to the texts, such as the fixed text reference. The evaluation of the dynamic reference in different environment produces a different set of direct text references.

8.4.3 Examples of Text References

The following examples illustrate the use of different methods to map the concepts of a D&T model with the texts. The examples are rather descriptive and should not be read

---

[62] Text references can be created also for concept classes. For sake of clarity the following text discusses about text references created for concept instances, although the same applies for the text references of concept classes as well.

literally. Most of these examples have been implemented with the current system and the rest are used as demonstrations.

Fixed References

The fixed text references are created by selecting one or more texts from the heading hierarchy that are attached to a new text reference. The model editor may create fixed references manually in the modelling environment. If an automatic classification system is available, it may associate a set of concepts to a text after the indexing phase. Similarly a NLP system utilising deep NLP analysis may perform the task. A set of view types must be attached to each of the references.

Word Baseform References

The conventional inverted file techniques can be used for creation of dynamic text references in form of pre-defined queries, as in several commercial products. The various levels on NLP can be used for producing content representatives that are used by dynamic text references. For example, results of morphological analysis can be utilised in the form of computational references via inverted index files.[63] Below is a dynamic text reference returning all texts, where the word "front" and "wheel" and "hub" exist in any inflected form. Thus this dynamic reference can be used to return texts related with front wheel hubs independently from the word forms used in the text. Use of word adjacency can be an useful extension for phrase retrieval although the ambiguities left to the texts after language analysis cause problems.

Baseform "front+wheel+hub"

Analytics References

The text references based on SIMPR indexing make an exception to the uniform outlook of the retrieval interface. They follow the two-phase retrieval paradigm adopted for the index phrase retrieval. In first phase all the index phrases containing the word(s) defined in the text reference are returned. In the second phase the user sees a list of index phrases, from which (s)he chooses the appropriate ones and the system returns documents where they were present. Alternative to the two-phase retrieval of a dynamic analytic reference we can convert the dynamic text reference into a fixed one during model development; the selection of phrases can be performed before retrieval. This means that a model editor evaluates the dynamic text references for each phrase reference and selects the phrases to be used with each text reference. The costs of this alternative include the amount of work to be done while selecting texts and the loss of feed-back from the display of index phrase list. The

[63] Here the example is in English although the morphological analysis is more relevant with inflective languages like Finnish.

example below causes the system to prompt the user with all index phrases containing word fuel.

Analytics: "fuel"

Structural References

Dynamic references can utilise the structure of the documentation. The frequent situation, where concepts referred to by the main chapter are referred to only implicitly by the sub-chapters and the following chapters, is one example of the cases where structure-oriented approaches to mapping creation are useful -- the concepts associated with a main chapter may be "inherited" to the sub-chapters and possibly also to proceeding chapters. Another common possibility is to utilise the classification hierarchy like structure of documentation. The informal example below describes a dynamic text reference that evaluates into a fixed reference pointing to a specific sub-chapter (or a set of sub-chapters) fulfilling the specification. I.e., an instance "GM V-8 AD" of the concept class *motor* is created a text reference to the sub-chapter(s) of chapter 5.5. whose headings contain the string "V-8". This kind of references can be created even before the documentation is available.

motor instance "GM V-8 AD"
-> find subheading containing substring "V-8"
under heading "5.5 Motor Types"
of document "Ford Granada"

Document Classes

The storage system used for the texts may enable manipulation of the texts as objects of document classes. In this case the dynamic text references may refer directly to named document classes and attributes tagged to the documents. This enables use of queries similar to traditional relational DB queries. Further, the separation of view types may be based on the class of the document referred to. The following example clarifies this view by presenting a dynamic text reference that refers to document objects in class "Installation Description Text" or any its subclasses:

motor instance "GM V-8 AD"
-> find texts in document class "Installation Description Text"
where product-code is "GM V-8 AD"
View Type = Installation

Combinations of Methods

The approach chosen in domain and task modelling system is an open-ended way to define text references. In addition to the methods described above we can apply methods based on vector space model as well as methods based on clustering and machine learning. The choice depends on the techniques available and on their applicability to the domain, documentation. and to the concept in question.

There exists also situations where a combination of methods would succeed much better in referring to the correct documents than any of the single methods [164]. A typical situation is such, where two methods in two dimensions support each other: in a previous example the inclusion of a term in the heading text was used and a restriction to a certain branch of the heading hierarchy tree can be used to limit the scope of an NLP-based index search. Typically even the use of fairly similar methods, f.ex. multiple query representations improves the retrieval performance [224].

8.4.4 Text Reference Classes

The second dimension of the text reference creation problem is the volume of the models. If a model contains hundreds of objects, the task for creation and maintenance of text references is still feasible, although requires a large investment in form of time used for the task. If the model consists of thousands or tens of thousands of objects, the hand-crafting can take several man-years[64] and is no longer feasible under normal economical constraints. For this reason, we have to pay attention also to the automation of text reference creation by following the artifact warrant principle.

This problem can be solved with techniques similar to the creation of a D&T model. Instead of creating text references for each stereotyped instance of a class we can define one single method for the class that is applicable for each of the objects. The methods used for creation of dynamic text references for all instances of a concept class are called *text reference classes*. Text reference classes are definitions of dynamic text reference methods that are applicable for all instances of a concept class. By defining one generic text reference method for the concept class we are able to define a dynamic text reference for each concept instance of the class at once. If we assume the size of the definition scheme to be two orders of magnitude smaller than the size of the instance world, the total effort needed for creation of text references can be reduced one to two orders of magnitude!

The methods defined in the text reference classes utilise uniform structural properties of the instances such as attribute values and obligatory references to related instances. They can access all information that can be expected to exist in a concept instance, to which the reference is to be attached. Thus the text references can be based on the attribute values of

[64] With average 6 minutes per text reference, 220 days a year, and 4 hours efficient working time a day the effort of one person is 8800 text references a year.

the objects but also on the related objects and their attribute values in the model. When several alternative mapping methods are available, the methods used for each concept class may be selected to fit its' special features.

The definition of the methods is similar to the definition of text references, including definition of the method used (heading hierarchy, baseform index, analytics...) and the definition of the argument. Instead of a fixed argument string used for the dynamic text references the queries contain a LISP expression to be evaluated in the context of the instance and it is supposed to return a string used for the actual retrieval. In the instance world the text references created by a text reference classes are visible as read-only dynamic text references attached to each instance of a concept class. In the retrieval phase there is no difference between the instantiated text references and the "normal" dynamic text references or even fixed text references. They all look alike.

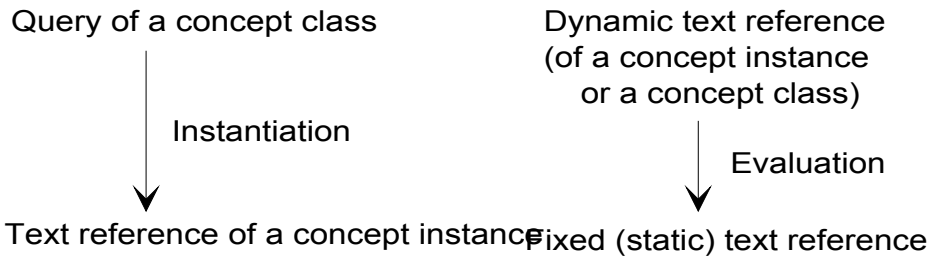

Figure 19. Instantiation and evaluation.

Below is an example definition of a text reference class. When this text reference class is instantiated for each of the instances of the concept class, each of them will have a dynamic text reference that utilise baseform indexes to retrieve all texts containing the name of the concept.

```
Baseform:  (attribute name *instance*)
```

As stated above, the text reference classes are not only references created based on the attribute values of a concept instance. The methods may utilise the full content of the model, the facilities provided by the LISP notation, and external data sources. For example, the following definition utilises an interface to external database and relations between instances:

```
Baseform:
(concatenate 'string
                (read-from-external-db
                        :db-name "products"
                        :returned-field "prod-code"
                        :key    "product-name"
                        :key-value (attribute name *instance*))
                "AND"
                (attribute "class-code"
                  (related-by "belongs-to-product-family" *instance*)))
```

This query uses a baseform index to return related texts for a product. The string used in the retrieval contains a product code retrieved from an external database and a class-code of a related instance combined by an AND operator.

In practice the definitions of text reference classes are simpler. For example the following definition that utilises document structure is a rather feasible method to retrieve descriptions of messages from a telephone exchange manual:

```
text reference class for concept class "message":
->      find subheading under heading "Message Descriptions"
        containing the number of the message in the title
```

With the document class definitions the example would be even more elegant; a document class defined for messages could be referred to directly by the concept class message and the selection of document could be based on message id.

A screen dump from the modelling system is in the appendix B. It represents the user interface for definition of text reference classes in a form window. The model editor enters two values to the form: the method used ("type of query") and the argument ("query form"). In addition the model editor defines if the dynamic text references instantiated from the text reference class are evaluated when created.

8.5 Workflow of D&T Modelling Process

As described above, the text reference creation process includes steps such as instantiation of a text reference class as dynamic text references of concept instances and evaluation of a dynamic text references to a fixed ones during retrieval (see Figure 19 above). Figure 20 presents the possible steps of a text reference creation process with a state transition diagram like formalism. The nodes of the net describe states of the work process and the labelled arcs starting from a node describe the tasks that can be performed when the node has been reached.

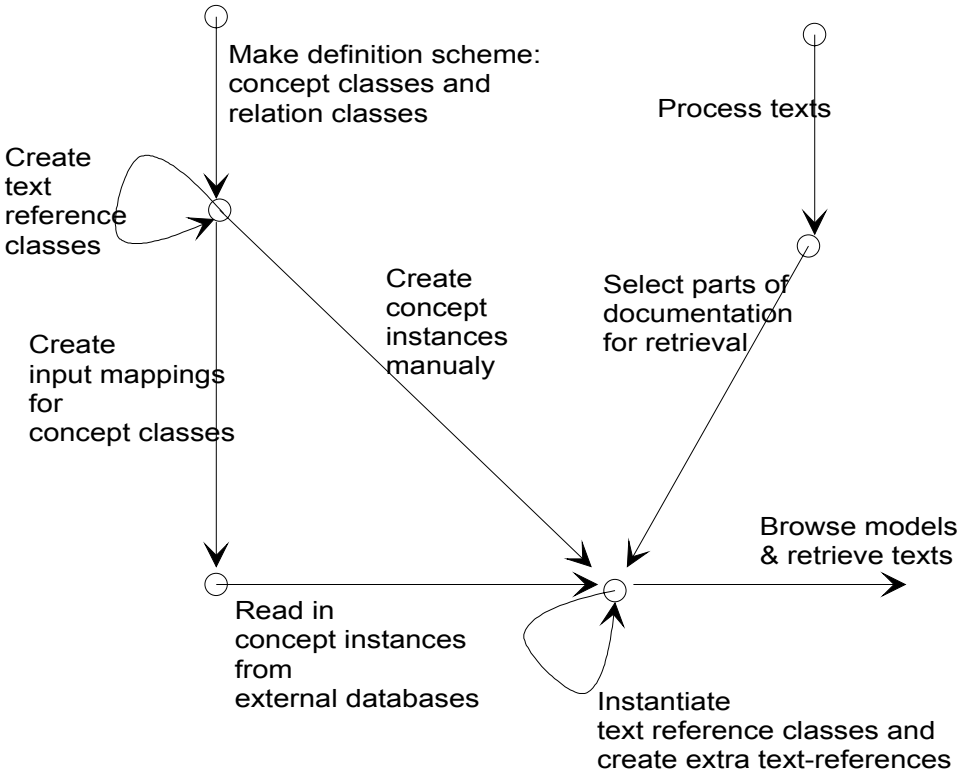

Figure 20. Workflow of D&T model and mapping creation.

The process can start either by processing the texts (on the right) or by manual construction of a definition scheme (on the left). When the concept classes and the relation classes of the scheme have been created, the user may choose one of the three alternative tasks: to define text reference classes for the concept classes, to make instances of the concept classes, or to define interfaces for design information input followed by reading in instances from external sources As soon as there exists some concept instances, the user may attach text references to them. Also the text reference classes can be instantiated to each of the class instances, as described above.

So far the user has manipulated only objects of a model. In order to make fixed text references or to retrieve texts the user has to select a documentation to be used. After this the dynamic references can be evaluated to fixed ones and texts can be retrieved. The user may also produce a read-only version of the text references by instantiating the text reference classes, evaluating all dynamic text references to fixed ones, and writing the model to a read-only format. This format can be used for retrieval of a fixed documentation selection with restricted HW environments, such as PC.

Version and configuration management are common problems that influence also the D&T modelling system. Both the models and the documents referred to by it change. When the documentation is large and complex processing, such as NLP, is used to extract content representatives from the documents, it is practically impossible to reprocess the documentation after updating a model. Similarly, updating (or selecting different parts of) a documentation may not require extensive changes or manual modifications to the model.

With the D&T modelling system the elaborate phases are encapsulated to some decree; documents are processed separately from the model building and the text reference creation used for mapping models to text is semi-automated. Fortunately this critical phase -- creation of text references -- can utilise the most effective and efficient methods available for each concept class, as stated above. Further, changes in external databases, where from the instance world is created, does not affect the definition scheme or require text processing, although requires re-creation of the text references. Similarly, changes to some concept classes force only (automated) input of the instances of changed classes from external databases and creation of text references for them. Thus the separation of the text processing from the model building and the use of object oriented models enables higher degree of automation in integrating text and design databases for creation of text retrieval systems.

8.6 Guidelines for Applying DTM

In this stage it is reasonable to summarise the features of the models and the automation methods proposed for modelling a domain with the DTM system. The following guidelines are based on the authors original ideas on applying the artifact warrant principle clarified by the results and experiences gained from the pilot (description starting from page 117). The reader is also recommended to return to this sub-chapter after reading the description of the pilot. The guidelines are valuable when a knowledge representation like the D&T model is applied to technical documentation retrieval. These guidelines are somewhat specific to the D&T modelling system, but the underlying view is of general form.

First of all, the major features of the D&T models relevant for the guidelines can be summarised as follows:

- Concepts with vocabulary and language independent identity.
- Multiple hierarchical relations between concepts.
- Non-hierarchical relations between concepts.
- Browsing and navigating relations.
- Views based on view type hierarchy for relations and text references.
- Two-way mapping creation, from models to text and vice versa.
- Static and dynamic mappings as text references.
- Combination of mapping techniques in text reference creation.

The DTM methods enabling efficient and economical application of the DTM in technical domains are mainly based on the two-level class-instance structure of the D&T models that is utilised in following features:

- Reduction of users cognitive over-load by defining explicitly what the model contains (e.g., closed world assumption).
- Semi-automated model creation based on design information input.
- Semi-automated creation of text references using text reference classes.

- Version and configuration management of large models.
- Searching for concept instances.
- Utilisation of document structures in text retrieval.

When the DTM methods described above are applied for text retrieval purposes we have several alternative approaches to model the domain. Many of these approaches and decisions to be made are similar to the decisions in thesaurus construction approaches or knowledge engineering methods, such as selection of the information to be included in a model, selection of the organisation of the model, as well as the way to represent tasks. Similarly, the interaction methods used for linking the models with the texts or text representatives, i.e., the creation of the mappings, can be realised by several means.

The information to be represented in a model should cover the central concepts of the domain via which the users can find the information they need. The existence of such central concepts is based on the general view in IR and philosophy. The level of elaboration chosen in some specific case depends on the trade-off between the coverage of a model and between the compact nature of the model. The coverage can be improved by adding concepts to the model, but it will hardly ever be complete. Instead, as the size and the complexity of a model grows, the average effort needed for finding the required concept, the storage requirements for a model, and the effort spent on retrieval and model building grow also. Thus the optimal size of a model should be estimated, and the casual information needs that are not supported by the model should be handled with some alternative methods. The availability of external information systems used for instance world creation affects also greatly the effort needed for model creation. Thus the models typically contain information that is easily available whereas hardly obtainable information is left out unless seen essential. In practice most of the models tend to contain too little conceptual information due to the effort needed for the creation of the models.

Second group of decisions has to be made for the structure of the model; What should be considered as a concept class vs. a concept instance, as a separate class vs. as a different attribute value, or as an attribute value vs. as a relation? These issues reflect the object-oriented nature of the models and some of the answers can be searched from the domain of object-oriented programming languages. Unfortunately, most of these answers are not quite appropriate for the domain models.[65]

The separation of concept classes and concept instances has great influence on the view taken to the modelling. Are the concept classes "the concepts" and the concept instances instantiations of the classes, such as "Mr. X", is an instance of concept class *man* or, are the concept instances "the concepts" and the concept classes merely a method to group concepts that share similar attributes or relations? Consider, for example, the motor example. Should

[65] For example, the rule "an object (i.e., a concept instance) should not change the class during its life" is rather irrelevant since the model is static and does not include processing that is visible for the user during the retrieval.

there exist a concept class *motor* with instances, such as "4/2,1/inj"[66] ? Or should each of these motor types be subclasses of the class *motor* thus enabling each of the physical motors to be the instances of the concept classes *4/2,1/inj*, etc.? Consider further integration of some existing subject classification scheme with other model information. Should the classification hierarchy be represented as a concept class hierarchy with no concept instances? Or should there exist one concept class, *subject classification*, whose instances form a hierarchy based on *subclass* or *broader than* relation in the instance world? In principle, all these alternatives are valid ones, and the answers depend on the chosen point of view.

The strongest guidelines for the use of the D&T models are merely present as definitions of the model:

- the attribute values are literal and relations are used for referring to other objects,
- each concept instance of a concept class shares the same structure (i.e., has same attributes), and is imposed by the constraints defined by the relation classes,
- each concept instance of a concept class can utilise the text reference classes defined for the concept class,
- each concept instance can be searched by its attribute values.

These constraints state merely when separate concept classes should be used instead of concept instances; if two concept instances of a class have different attributes or they have relations that are not defined by same relation classes, they should be defined under different concept classes. After these constraints the guidelines are based on the results: what kind of models are easy to use and can be created as well as linked to the texts with reasonable effort. From this point of view it seems to be preferable to minimise the number of concept classes to those that are required to be different ones. Smaller definition scheme helps in creating an overview of the domain and thus makes the retrieval process faster. It also reduces the model creation effort and makes the maintenance of consistency easier. From this point of view the modelling in definition scheme is reduced to the definition of minimal features (different concept classes, attributes, and relations) needed for the modelling in the instance world. Thus the term "concept" is used here as a synonym of "concept instance". Thus the preferred answers to the questions stated above are: Do not use concept classes for motor types since the individual motors are useless as instances of these types during document retrieval. Prefer use of one concept class "subject classification" to the use of separate concept classes since there is no need for that, i.e., neither different relation classes nor attributes.

---

[66] Which is a motor type, whose cylinder count is 4, volume is 2,1 liters, is and injection motor etc.

A closely related issue in conceptual modelling systems is, what kinds of relations should be used in a model. As stated above, the model should include information that enables efficient text retrieval. Thus also the choice of relations is domain-dependent. When we recall the description for an ideal knowledge representation for text retrieval purposes -- as stated in the framework -- the concepts should represent the mental images and domain structures string-independently. In our model they are represented as interrelated objects with attributes. The thesaurus relations, like synonym, are relevant for terms; two objects describing mental images cannot have a synonym relation although their textual representations could. Thus also the use of synonym relations between concepts describing the images should be avoided. Instead, two concepts may be mapped to the same text representations, may have other relations to (almost) same set of other concepts, or may even have same attributes and attribute values used as names, etc.

The next selection is, how to describe tasks. The DTM system offers two mechanisms for this: the concepts and the view types. Using a concept class "task" a modeller can build representations analogous to goal hierarchies, scripts, workflow diagrams, Petri nets, etc. The most attractive approach seems to be the use of goal/task hierarchies, where each concept in a hierarchy represents a goal or a task that must be satisfied or performed in order to achieve the goal on the next higher level of the hierarchy. The hierarchy is formed using *subtask* or *subgoal* relations. In addition, each of the concepts is linked with the texts describing the task and has relations to the other concepts that are relevant in performing the task. This approach enables different user groups to follow different levels of detail in the instructions: novices can follow all subtask relations whereas the experts use the top levels as a list of main tasks. Alternatively the tasks can be represented using the view types, i.e., by defining a view as the subset of relations and text references (and concepts and texts connected by them) that are needed for accomplishing a certain tasks. Based on our current experiences with the system a combination of these two methods seems to be the most appropriate solution although no specific guidelines can be given yet.

From a procedural point of view there are two directions to link a model with the texts and text representatives. During text processing each text to be included in a database can be linked with concepts of the model, like in manual or automated text classification process. From the other point of view, these mappings can be established by defining the texts referred to by each concept. If the creation of the linkage requires heavy processing, such as some classification systems or NLP, it is natural to create the mapping during the processing phase instead of retrieval time. This also encapsulates the addition of a new text to the text itself. The advantage of the model oriented creation of the mapping is the ability to change the mapping when the model changes without reprocessing the whole text database.[67] The

[67] In both cases the resulting mapping can be stored with the texts, as well as with the model, or even with some other storage structure. This mapping can also be accessed from both directions, to find the concepts described in a text or to find the texts describing a concept.

combination of processing and model oriented creation of the mappings can avoid most of the problems of these two approaches. The text representatives are created during a (possibly elaborate) text processing and the final mapping is created from the direction of the models, via text representatives. Thus the addition of a text changes only the text representatives, the changes in a model are local, and the creation of the final mapping is reasonable fast and can be performed during the retrieval.

9 EVALUATION OF PROPOSED SOLUTION IN A PILOT

The previous chapters presented the DTM system as such, interaction between the models and the users, as well as the creation of the D&T modes with the DTM method. This chapter focuses on evaluation of the previous work. We start with a description of a large pilot case, a text retrieval system (TRS) build for a technical domain. where the DTM system is used as an integrated component. Then we make a couple of contrasting examples by applying the DTM approach to two much smaller domains, on two paper-based cases. On this basis we continue with the evaluation of the artifact warrant principle, the DTM methodology, the DTM system, and its' integration capabilities.

9.1 TeleSIMPR Pilot

The DTM models and the DTM domain modelling approach described above were tested with an industrial pilot case at Nokia Telecommunications producing switching systems and other telecommunication equipment. A pilot system called TeleSIMPR was built for processing and retrieval of 20 000 pages of customer documentation describing DX 200 digital exchange. Earlier this paper documentation has been delivered in 70 folders causing problems typical for paper documentation (cf. pp. 68-) and even occupying more storage space than the exchange equipment itself. The experiences described below are based on tests with a portion, about 10 MBytes (i.e., 5 000 pages) of documentation. The users of this documentation include both operators of telephone exchanges and designers producing a new product generation. Many of them would like to browse a model describing the relations of the software and hardware components of the exchange, such as modules, programs, alarms, and commands. The purpose of the DTM based retrieval system is to provide them with a compact model and to link the texts to the model.

9.1.1 Document Processing

The Figure 21 describes architecture of the pilot system. The documents processed using a SIMPR-based text processing system go through several phases preprocessing and natural language processing (including morphological analysis and disambiguation), and classification. During preprocessing the chapter - sub-chapter structure of documents is extracted and stored as a table of contents called a heading hierarchy. After preprocessing the documents are manipulated in smaller units called texts, which in the SIMPR terminology refer to a heading and a (possibly empty) text up to the next heading.

After processing the texts can be linked with the D&T models with a variety of methods. In our case the possible strategies include: use of inverted files with no NLP, use of word baseforms (and compound words) as index terms after morphological analysis, use of disambiguated word baseforms, use of automatically extracted multi-word index terms,

and use of tree structured index terms. Each of the latter possibilities gives improved functionality but requires some extra processing. In the pilot domain we found the use of word baseform queries to be sufficient for most of the cases. This means that the names of the concepts, their abbreviations, or some other attribute information is, in most cases, adequate for identifying the existence of a concept in a text in this domain. Although this can bee seen as a consequence of the special features of LPS documentations (cf. pp. 73-), it would not be justified to generalise the observations of this one pilot case to other LPS domains.

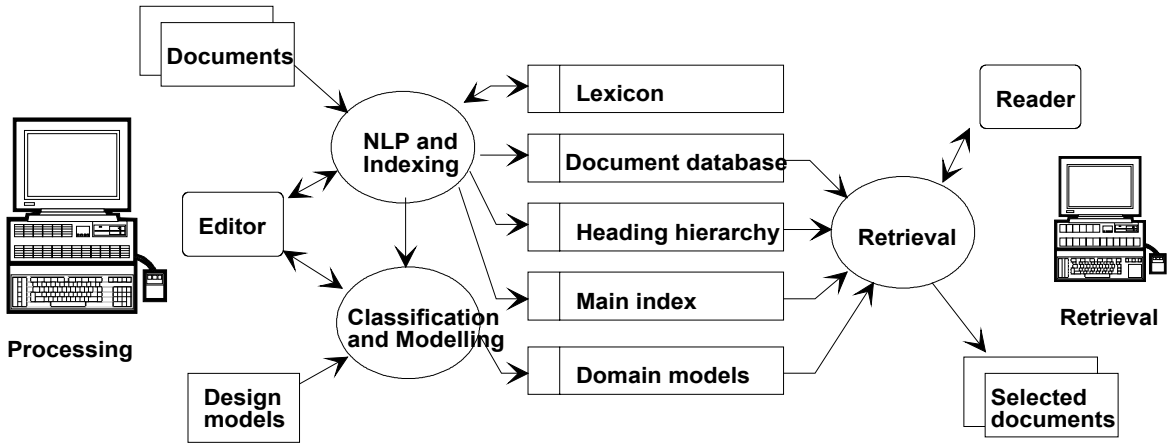

Figure 21. Architecture of the TeleSIMPR pilot.

The techniques developed in SIMPR are domain and language independent. By this we mean that the language-dependent parts can be implemented as separate databases: lexicons, rule sets, etc. The current implementation processes English and Finnish. When applying the NLP to a particular domain we have to build the domain-specific parts. In the case of DX 200 documentation, 4 000 new entries were added to a general lexicon containing 55 000 entries (interpretations including word baseforms and other lexical information). The effort needed for adding 4 000 entries was minimised by semiautomated lexicon compilation tools, which extracted entries from data banks maintained by the organisation. They included domain-specific terms, reserved names, and abbreviations. More than 95% of these were either proper names or other nouns, which simplified the manipulation. The manipulation of incomplete sentences is solved by the inherent features of constraint grammars [107] aided by internal mark-up. So far we have not made other domain-specific changes to the processing although the language analysis and index term extraction could also reflect the peculiar and restricted usage of language in technical documentation.

One general problem with NLP systems is the processing speed. The prototypes made in the SIMPR project do not make an exception, the early versions of the NLP and especially of the index term extraction system were unoptimised and rather slow. For this reason we have to estimate the benefits and costs of processing at each level of processing in each phase of the work. In order to offer a uniform retrieval interface, we can hide the methods

used with the domain and task models; each of the indexing methods can be used as a way to link the concepts in a D&T model to the texts. These processing speed requirements enforce us to use an efficient workstation in the text processing phase, whereas the retrieval environment has to be some less advanced, inexpensive HW/SW environment with a slow optical disk, where also casual users are able to use the system. This means that the functionalities of the retrieval environment should be minimised, the access times should be optimised, and the user interface should be easy to use. The approach adopted here is to use separate processing and retrieval environments and make conversions between them. A conversion from a separate processing environment to a delivery environment means, on one hand, extra processing, but on the other, it provides a facility to optimise the data structures and access times.

9.1.2 Domain Modelling

The major purpose of the pilot was to evaluate the requirements stated for KR supporting technical documentation retrieval, the D&T models, the semi-automated modelling techniques following the artifact warrant principle, and the DTM system implementation. Our role was to aid in the use of the DTM system and to observe, how naturally and efficiently these tools and underlying ideas can be used in the pilot. Thus the creation of the domain model was performed mainly by the experts of Nokia Telecommunications, who collected information needed for the model. With our help they built a model describing the DX 200 domain, extracted the data for the instance world from their information systems, converted the data into an instance world as well as created text references for the instances. Much time was spent on collection of relevant information and on structuring the definition scheme of the model. The domain is fairly large and information had to be collected by interviewing several experts, who focused on separate parts of the system and gave information from different points of view. The process was similar to a knowledge engineering task, where the operation with the tools takes only a small portion of the time as vast majority of the time is spent on collecting information and on considering alternatives extracted from alternative sources of information. From this point of view the model reflects the "collective view" or "common knowledge" of the designers of DX 200 system. As this view changes, the model should also be changed. Especially a model describing a new product generation under development will go through several stages of development until it stabilises to a rather fixed view to the product generation.

After the "design" of the model was finished (i.e., the definition scheme was build) the next task was to collect data from existing information systems. For this purpose the experts of NTC extracted information from relational databases and filtered data from definition files by special programs. Consistency checks were performed for the extracted data, especially when the attribute values for an instance were collected from several sources. Creation of the actual instance world was fairly straight forward. The extracted data files

were read into the model with the conversion routines within a couple of days. The size of the transmitted data files varied a lot largest instance files containing several thousands of instances. Some concepts' instances were also created manually, since this information was not stored in any database format. The simplest way to do this was to create manually an input file for the conversion program.

Creation of the text references was performed within a few days. For each of the concept classes the experts defined methods, that were applicable for all instances of the class. The references were tested with selected individual instances before applying them to all instances of the class, and some iteration was needed before the final versions were defined. Most of the references were created using simple index searches on a selected attribute of the concept, such as a number or an abbreviation of an alarm (possibly in a combination with the full English or Finnish name of the concept). The advantage of the abbreviations is that they form a sort of controlled indexing vocabulary, although also a couple of ambiguous abbreviations were found in this domain. Unfortunately they are mnemonic and hard to remember and the documents do also use the long names instead of numbers and abbreviations. In the pilot the texts containing the full names of concepts are found simply by searching documents containing the adjacent words of the concept name.[68]

In this domain the structure of the documentation is deeply classification oriented. In the pilot there were two alternatives to utilise this feature, either explicit or implicit, to create different views to the documentation. The explicit alternative is to create different views based on different text references of a concept referring to different parts of the documentation. For example, when the users want to access only a single text defining a concept, they can use the text reference (included in *definitions* view) that access only the part of documentation defining the instances of this concept class. Another text reference of the same concept accesses all occurrences of the concept in all parts of the documentation for the users willing to see them all.[69] In the pilot we chose the implicit views: The pilot users familiar with the documentation can make the distinction between definitions and other occurrences based on the heading information of texts referred to by the concept, i.e., based on the part of documentation and the name of the document listed for each text referred to by the concept.

Figure 22 displays a part of the definition scheme of the pilot. This figure is taken from the screen of a PC retrieval interface, where the scheme was drawn manually after the first tests. The blocks on the screen display concept classes that were found to be relevant for retrieval of software oriented parts of the documentation. The relation classes are displayed as labelled arrows connecting the concept classes. For example, each concept of the class

---

[68] In this domain, the existence of term banks enables us to recognise them also during the language analysis, i.e., text processing phase.

[69] In some cases the facility to see an exclusive list of the related cocepts is expected to reduce the need for searching for all occurences of a concept in the documentation.

Program Block (in the middle) is a *part of* (arrow labelled with P) a *Service Block*, *uses* (U) *Messages* to communicate with other program blocks, *uses* (U, i.e., reads or writes) *Files*, *uses* (U) *Alarms* that can be (*ISA*) *Notices*, *Failures*, *Disturbances*, or *Diagnosis Alarms*, etc. When the concept classes, such as the *Messages*, have several thousands of instances, the required one is accessed by searching suitable attribute values. For example, by clicking the *Message* concept class, the user can request for a *Message* with a given *number* attribute, or *Messages* whose *Finnish name* or *English name* begins with a given string, etc.

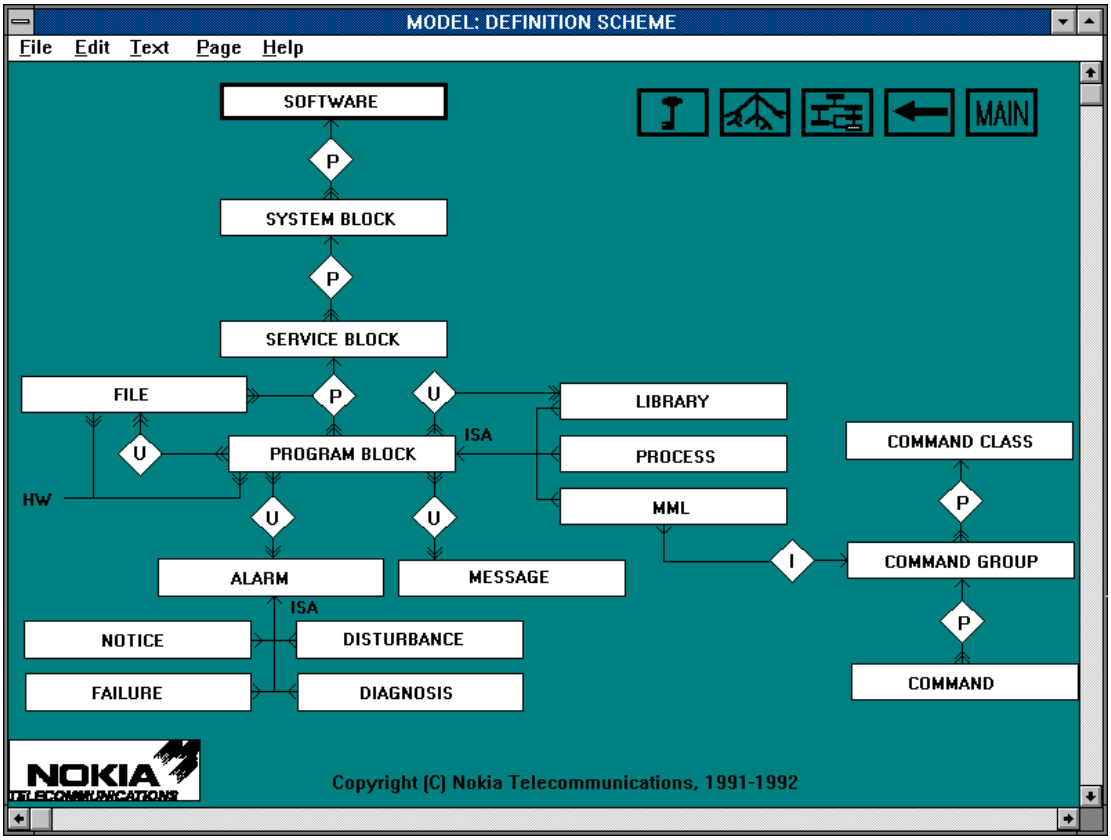

Figure 22. A part of the definition scheme in the pilot application.

9.1.3 Text retrieval

The use of the models in this domain is exemplified by the following story: An operator of an exchange receives a notice print-out on the console reporting a failure to read a (unspecified) parameter file. How to get information about an unknown entity? Using the known information, the number or name of the Notice, the operator is able to access the right Notice concept from the model. By following the uses relations the operator is able to find the Program Block(s) that sent the Notice, further, the File(s) used by the Program Block(s), and finally, the requested documentation. In real life this kind of navigation chains contain only a couple of steps, but cannot be performed without knowledge about the conceptual structure of the domain, i.e. cannot be substituted by term co-occurrence search or other similar text representation level techniques.

The usefulness of the D&T models in text retrieval varies according to the users and their needs. From the text retrieval users the knowledge intensive parts of organisation, such as the system testing department, use most the kinds of information represented in the models; the engineers testing a new system configuration use exhaustively all the documentation of the system. According to an internal user investigation of the pilot organisation the system designers and experienced operators will appreciate the full facilities of the system whereas some operators on the field use only operating manuals that can be accessed fairly straight-forward using a single view to the documentation -- a table of contents listing the commands, alarms, and messages.

The overall structure of the domain can be browsed more easily by a compact model than using textual documentation. Typically the test users first wanted to browse the D&T models as such, without retrieving documentation, and only after that checked that relevant documentation related to some known concept could be located. Usually they wanted to compare the model with their mental images of the domain that was build piece by piece from documentation, from seminars, and from conversations with colleagues. The model and the mental images were sometimes contradictory. Sometimes the users found a piece of information that they had been looking for earlier without finding the answer. Thus the models were appreciated since they describe the structure of the system and give context to each individual concept and piece of information. From this point of view the D&T models were recommended to be used especially for self-learning as well as for supporting seminars for new employees.

9.2 Case 2 - IGE Software Library

This sub-chapter describes case 2, modelling of a reusable software library. This case searches limits of the DTM modelling approach in a much smaller domain than the domain in the pilot. The case is handled on paper basis, i.e., no implementation takes place.

9.2.1 The Domain

IGE (Interactive Graph Editor) is an in-house software library containing reusable C++ classes. It is used for writing applications that manage graphs and graph-oriented graphical user interfaces (GUIs) in Microsoft® Windows™ and Motif® environments. IGE consists of about eighty C++ object classes, mainly based on classes of another (commercial) development environment called C++/Views® (by Liant).

IGE documentation is organised as a manual containing about 200 pages. The manual is divided into three parts: the User's Guide, the Technical Reference, and the Class Reference Manual. The User's Guide aims at giving insight to IGE, whereas the Technical Reference is meant to give constant support to the application developer, and the Class Reference Manual contains detailed descriptions of all classes. Readers of this manual (and especially of the Technical Reference) are expected to be familiar with object-oriented programming

in the C++ language and to have a basic knowledge of the C++/Views class library. C++/Views will be described only to the extent necessary for understanding IGE.

The structure of the domain is characterised by the class hierarchy of IGEClasses. Each class (e.g., IGENode, IGEEdge) inherits some functionality form parent classes and implements some functionality themselves. Most of the classes inherit functionality also from C++/Views classes. The classes may be included in a containment hierarchy, i.e., CompoundImage may contain ImageComponents that may contain an Image or IGESlot is included in IGESlotGroup that is included in IGEViewNode etc.. The ImageComponents (e.g., HotSpot, IGEJoint) take care of the physical display operations (e.g., flips, zooming operations).

From another point of view the library is organised to layers according to expanded MVC paradigm (Model, View, Controller). Classes fall on 6 different layers according to their functionality (e.g., Event handlers on Controller layer, IGENet on Model layer). Member functions implement the functionality. Functions include initialisation, update operations and policies, visualisation (display) operations, such as rotation, rearrangement, focusing, selecting, activation, validity checks, mouse events, repository functions etc. The user tasks for building applications with services of IGEClasses are also described in the manuals.

9.2.2 Modelling IGE Domain

Modelling of the IGE domain is performed here on paper basis in order to illustrate the strengths and weaknesses of the DTM approach. According to the guidelines given in previous chapters, the modelling starts with creation of the definition scheme. The figure below illustrates a definition scheme created for this domain based on browsing through the manuals and discussions with experts of the domain, i.e., the main designer of the library and an experienced user of the library. The notation used in the figure is oriented towards OMT notation due to the availability of suitable graphical tools. The names of attributes and relation classes are represented as well as the multiplicity of the relations (- = one, o = 0-1, • = 0-n), but direction of the relations could not be presented by this tool. The main user interface for a D&T model based text retrieval system for IGE documentation could use similar outlook.

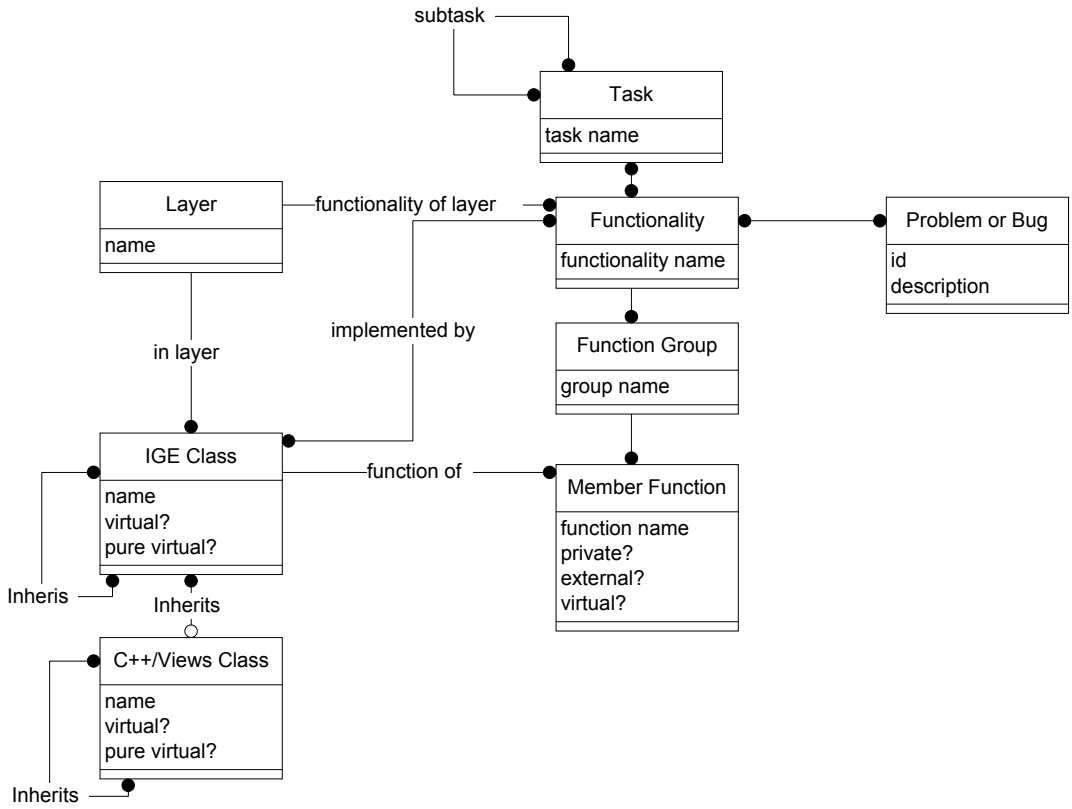

Figure 23. A draft definition scheme for IGE document retrieval.

The next two steps in DTM are

1. definition of converters from existing information sources to automate the creation of instance world and/or manual creation of concept instances, and
2. definition of text reference classes for the concept classes.

The creation of instance world can be automated partially, as follows:

- The 80 IGEClasses, their inheritance relations, containment relations can be acquired from the header files by using simple scripts or services of CASE tools.
- Member Functions, their attribute values and the "function of" relations can be acquired as above.
- C++/Views classes and their relations can be acquired from the source files of this commercial library.

Following information has to be hand-crafted:

- The architectural Layers (6) and their relations to other concept classes.
- The Functionality concepts representing functionality provided by a combination of classes (e.g., creating an icon representation to the screen) are rather abstract. They and their relations to function groups and tasks are not represented formally, but only in natural language.
- Problem / Known Bug concepts are not represented in a form suitable for automated input.

Some parts of the instance world can - in principle - be acquired semi-automatically, but is, in practice, best to hand-craft:

- Most of the Tasks and their "subtask" relations could be extracted from the documents where the subtasks are typographicaly marked (as numbered lists). In practice, manual creation of the instances is easier if the modelling process does not re-occur often due to version development.
- Function groups could be extracted from C++ header files, if they were tagged uniformly.

The text reference classes useful for the IGE model could be as follows:

For all concepts

- A reference to all occurrences of a name field ("class name" "task name" etc.) in the documentation.

IGEClass

- A reference to the Reference Manual to the section containing the "class name" attribute of the IGEClass in the heading.

Task

- A reference to the User's Guide to the section containing the "task name" attribute of the IGEClass in the heading.
- A reference to the User's Guide to numbered lists (i.e., sub-task lists) containing the task name. (This could also be removed as the task hierarchy is available in the model.)

C++/Views Class

- A reference to the User's Guide section describing C++/Vies library the chapters containing the "class name" attribute.
- A reference to the C++/Views documentation, to an appropriate chapter heading, if the documentation of the external library is available in electronical form.

At this phase it is possible to define different views to the model. One possibility is to use views to represent different tasks, e.g., installation, library learning (overview), application building, debugging, and library maintenance. The alternative is to simplify modelling by leaving this additional means for model structuring unused - the model is probably small enough to be managed without view mechanism. The concept "Task" and its "subtask" relation is a sufficient means to represent the central tasks of the domain.

9.2.3 Advantages and Drawbacks

In this case the artifact (i.e., the IGE library) and the design information are almost the same. There is not much other design information than the library code itself. In case of software the artifact is often (unfortunately) said to be the best design description and documentation of itself. The bright side of the story is that the structure of the domain can

be extracted from the system itself. Thus the role of the domain model is to distil the central domain structures - making the modelling relatively easy. From the model use point of view the role of the model is to visualise these central structures to improve the efficiency of documentation use.

The concrete concepts, their attributes and relations can be extracted from the software relatively easily. Abstract information, such as tasks in using the library, are documented -- more or less -- in textual form. This reflects the status typical to SW documentation: the concepts can be extracted rather efficiently by semiautomatic means even in small domains / documentations, but the tasks and the functionality from the user view need to be hand-crafted. From the point of view of resource utilisation the DTM approach seems to be valid; it is best to start from the conceptual domain model that typically exists in suitable form. The task oriented information can be added to this basis, if / as soon as it is available. In general, the model creation part of the DTM approach can be applied rather well to this domain.

Due to the small size of the IGE domain no official naming policies, identifiers or other formal means are applied to the design. The use of C++ language has though enforced use of a kind of "normalised naming" along the C++ classes that is visible also as a naming policy of the domain and documentation. This naming policy enforces a simple mapping between names and the major concepts of the domain. Thus simple name based searches are often sufficient for locating the occurrences of the concepts from the documentation. Although this principle holds for majority of the documentation consisting of programming language entities, task names and other identifiers for non-program-entity concepts vary within the documentation. In general, the text reference creation phase of the DTM modelling seems to be reasonably well applicable to this domain.

From the end-user point of view the D&T models are likely to succeed in their intended purpose - in displaying central concepts and their relations as well as to link the concepts with the documentation with reasonable effectiveness. But this is probably not enough in this domain. This domain is oriented towards software re-use. This means that a real implementation of a text retrieval system should not only retrieve documents, but preferably also retrieve the appropriate sections of source code. Further, the links to the other libraries used (e.g., C++/Views) and to the other tools (e.g., compiler & debugger environment) may require support from the user interface.

In a comparison to the DX 200 case this IGE case gains less from the DTM modelling methods. From the positive side, the result is useful in reviewing the documentation. Further more, browsing the model is useful especially for getting information about the domain as such. The negative side is that the effort needed for the implementation does not give sufficient pay-back. The size of the model is rather small reducing the advantages of automation. Further, the number of users is rather limited reducing the effort available for building retrieval tools of any kind. In this case the document processing and the modelling

both take some effort, but especially tailoring the end-user interface and integrating it with the user environment would require substantial effort preventing the implementation of the scenario presented here. This comes back to the issues discussed earlier - the importance of evaluating TRS approaches in the scope of organisations needs.

9.3 Case 3 - Visual Planner Graphical Tool

This sub-chapter gives further clarification to the applicability of the approach by describing case 3, modelling of a graphical project management tool. The size of this case is somewhat larger than the size of case 2 - this manual includes round 300 pages - but is still comparable to it. Also this case is handled on paper basis, i.e., no implementation takes place.

9.3.1 The Domain

Visual Planner® is a project planning and management system for the Microsoft® Windows™ environment. It uses graphical approach by allowing the user to interact with the project's data in a form close to a pen and paper paradigm. Same time it provides computerised data processing and storage. The tool views a project database through activity nets, Gantt charts, and resource views. Running in a windowed environment it uses extensively windows, pull down menus, dialogue boxes, and graphical manipulation of object representations using mouse.

The central concepts of the tool include project objects representing the real world entities and (e.g., projects, activities, dependencies, resources, and schedules) view objects (e.g., timescales, todayline, crosshair). These objects are manipulated by various tools (e.g., dialogs) and operations (e.g., moving an activity in a timescale) launched from menus or activated from graphical representations of the objects. Different operations and tools manipulate different data fields of the project objects; moving an activity changes its starting time while its resources are changed by a dialogue.

9.3.2 Modelling the Domain

Figure 24 represents author's view of the domain as D&T concept classes and relation classes. The notation adopted here is due to the available OMT tools, as in case 2 (◊ = D&T subclass relation).

The creation of the instance world would take place mainly by hand crafting. Most of the concepts classes have only about ten instances, e.g., "View Object", "Project Object", and "Menu" have each round ten instances, "Views" only three. There are still a couple of concept classes with tens of instances, that can be extracted from the source code, e.g., "Menu Command" and especially "Data Field". The number of instances in other concept classes fall in between these two groups. In general, the number of concept instances in each concept class is smaller than in the IGE case, and also less effort can be saved by

automating model creation. In this case the main justification for automating the modelling process should be found from improved consistency between the system and its model, which is useful especially when there are several versions of the product.

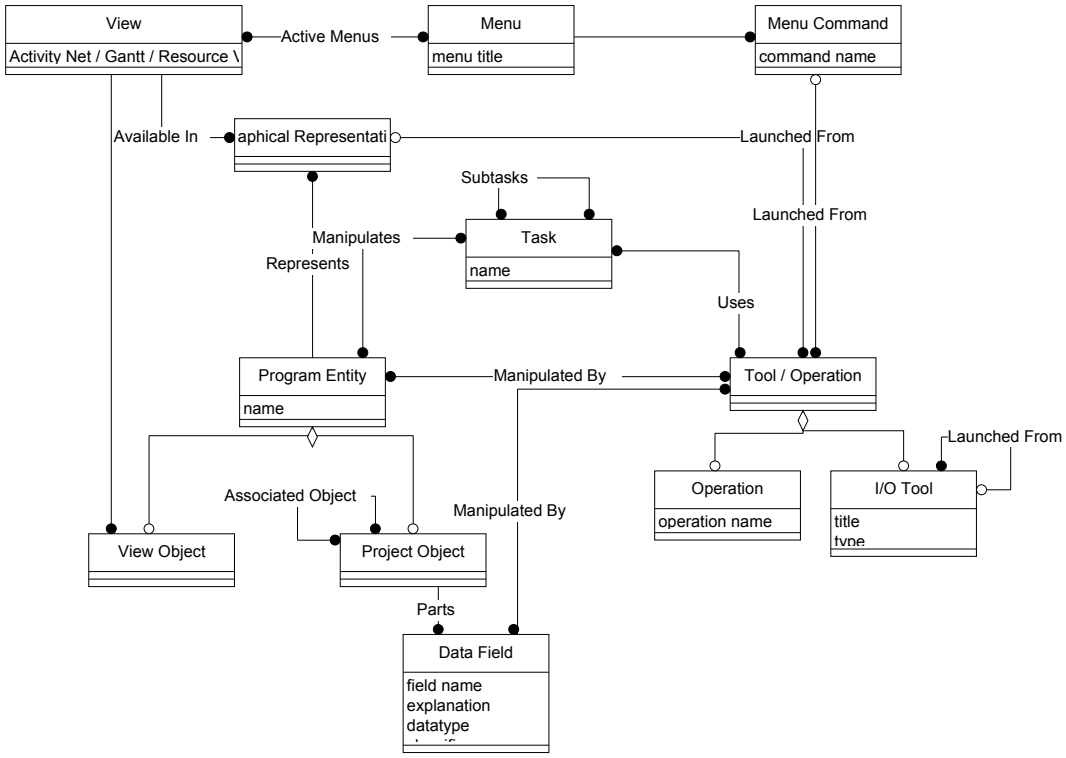

Figure 24. A draft definition scheme for Visual Planner documentation.

Creation of text reference classes is rather analogous to the IGE case. It is relatively simple to define text reference classes of form "find the subheading containing name attribute of the instance under the main heading XXX" for concept classes "Menu", "Menu Command", "I/O Tool", "Project Object", "View Object", and "Data Field". Also searching all occurrences of the terms, such as "schedule", can be useful as the term is unique and the documentation is small.

9.3.3 Advantages and Drawbacks

A natural form to execute a D&T based system with Planner would be to use it as an on-line help system. In this case - as well as in the IGE case - the domain is small and finding the text for a known concept instance is relatively easy with or without the text reference classes. The focus in modelling is more on providing means for communicating a overview of the domain to the user than in finding sophisticated means for improving text retrieval effectiveness in accessing the documents via the model. Altogether, we need to pay considerably more attention to the effort needed for and opportunities provided by the integration to the end user environment than in case of a typical stand-alone TRS.

Both in this case and in IGE case the author chose to use concept class "Task" to represent the user-oriented tasks in a uniform way with the rest of the model. As these cases have not been implemented, we have not observed of the value of this approach to the end user. What we can say about task representations is that the information needed for representing tasks is typically not available as formal representations like most of the information needed for the other concepts.

One of the main differences between the IGE and the Planner cases is the larger average number of concept instances in each concept class in the IGE example. One probable reason for this is that the IGE domain itself exists mainly within a limited formal world, i.e., within a programming language. As a contras, the Planner includes a kind of meta model of the project management domain and thus operates on a higher level of abstraction. Thus also the concepts of Planner seem to be rather specialised. Note also that in this case the use of inheritance prove to be useful for simplifying the model[70].

Choosing the level of abstraction is problematic in Planner domain. On one hand, the "Project Entities" and their interrelations are interesting to the user. They form the core model that represents the world within this tool, i.e., projects, activities, resources etc. On the other, the main purpose of DTM is to represent the domain of the documentation, i.e., the tool rather than the model within the tool. Thus the problem is; should the domain model represent the domain (i.e., the Planner tool), the model within the domain (i.e., the Project Entities), the domain represented by the model within the domain (i.e., real-world resources, schedules etc.), or a some combination of all these?

The more levels are represented explicit by separate concept classes the less there are instances per classes. This leads to flattening the two-level modelling into a single level model like in the thesaurus approach. The structurality and simplicity of the top-most level is lost and the model does not provide means for automating the modelling or text reference creation task. The model constructed for Planner aims in communicating overview of the domain by simplifying the definition scheme. What is lost is the second level, i.e., the specific relations between the "Project Entities" that now are just related by "Associated Object" relation. Author's view is that this information should be represented for the users as a separate model, but not necessarily by the D&T models. Implementation with D&T models would require the instances of concept classes, such as "Project", "Activity", and "Resource", to be meaningful for text retrieval. I.e., there should be documentation describing individual projects, activities, and resources (e.g., "Building World Trade Center to Helsinki", "Electricity design of floor 4" and "Machine 123").

---

[70] Relations between subclasses of "Program Entitiy" and subclasses of "Tool/Operation" manipulating them.

9.4 Evaluation of the Approach

9.4.1 Views to Evaluation

The purpose of this chapter is to evaluate the ideas presented earlier, i.e., to evaluate the artifact warrant principle guiding domain modelling for text retrieval in LPS documentations, the modelling approach derived from it, and the requirements stated for KR in general and especially in these domains. This evaluation is mainly based on the experiences gained in applying the DTM system to the large pilot described above, whereas the two paper-based cases are used mainly for estimating the scaleability of the approach. The reader should carefully distinguish the results that, on one hand, are based on the underlying ideas, and on the other, are dependent on the DTM system implementation or reflect special features of the pilot.

The focus on this evaluation is on domain modelling rather than on text retrieval. Our intention was also to test quantitatively the text retrieval efficiency gained using the D&T models. This would have been useful especially from the IR research point of view. Unfortunately there were two obstacles. First, there seems not to be metrics mature enough and applicable for measuring the dimensions relevant for technical documentation retrieval.[71] Secondly, the pilot organisation did not consider the quantitative information to be worth their resource allocation needed for the test. They considered the qualitative evaluation and response from demonstrations, such as Telecom'91 at Geneva [216], to be sufficient for committing on the technology. Thus they commercialise it as an integral part of their product documentation and announce the first DTM based products in the near future.

9.4.2 Artifact Warrant Principle

Altogether the experiences from the pilot are well in accordance with the artifact warrant principle. Extracting information from other information systems is feasible and the extracted information forms majority of the conceptual information needed for text retrieval. As described above, the concepts used for text retrieval in the pilot study are mainly functional or logical parts of the system, especially software components. The test users preferred navigating these concepts and their relations to full-text searching in most cases. Experienced users did also use the "table of contents" of the whole documentation for locating familiar documents describing known concepts. In the model (English or Finnish) names, abbreviations, or other identifiers of the concepts are in most cases used as the mapping from the concepts to terminology. Similarly structure conventions of the documentation are utilised -- explicit or implicit -- in mapping the concepts against the documentation. The availability of useful information in external information systems is

[71] This issue will be addressed further in the chapter "Future Research".

rather good at the pilot site and less than a percent of the instances or relations were created manually.

The problems in modelling are mainly faced in integrating the semantics of representations from various separated information systems to a uniform and compact single representation needed for efficient text retrieval.[72] Consider, for example, integrating versioned and non-versioned information from two different systems. In general, the pilot displayed a growing need for version development and configuration control enforcing tighter integration between the domain modelling system and other information systems. Also this emphasises the need to have homogeneous semantics among all the information systems; to make the exchange and integration of information possible in the domain modelling system. Thus the existence of structural information in external information systems is not always sufficient, as the artifact warrant principle suggests, but the homogeneity of the semantics of the information must also be considered.

As described in the context of the framework for TR representations (and in the previous chapter), world and domain modelling suffer from problems due to the magnitude of the knowledge to be acquired and represented. This means that the artifact warrant principle is applicable merely in domains, where most of the stereotyped features, entities, and relations of the domain can be imported from exists databases, i.e., technical databases, customer and product databases in commercial organisations, student / course databases in universities, etc. Essentially, only organisations with information in some well structured and defined format can apply the principle.

This work has focused on technical domains where the availability of external databases is rather good. Increasing complexity of systems has enforced organisations to structure their product information. A contemporary technical organisation without sophisticated information system is unlikely to fulfil its' mission successfully if it is unable to manage the internal information as well as satisfy customers' demands on high quality and reliability. In the near future the vendors of large technical products and projects have to either give up the business or prove their ability to manage their products and quality by ISO 9000 quality system certifications or other similar means. In case of software quality standards this means that the organisations are explicitly enforced to manage things, such as the products, elements of products, versions, configurations, relations between product elements and documents, etc.[73] In practice, this means that majority of the technical organisations producing large documentation to their large products or projects has the information needed for applying the artifact warrant principle now or in the near future.

---

[72] These problems are addressed by multidatabase research, although the integration of structure -- like the domain models -- is considered to be less problematic than the integrtion of behaviour or constraints. For further, see proceedings of CIKM-92 [246].

[73] See ISO 9000-3, Software QA, for further. It enforces organizations even to "collect, index, and store quality records". ISO = International Organization for Standardisation.

9.4.3 DTM and KR Requirements

Next we shall focus on the requirements for KR stated earlier (p. 81) and inspect two issues in parallel; 1) does the DTM approach and/or the pilot system satisfy the requirements and 2) are the requirements relevant from the point of view of the pilot.

The *expressive power* of D&T models is sufficient for representing the central concepts of the domain as well as *multiple hierarchical* and *non-hierarchical relations* between them -- although only binary relations are allowed (see p. 137). The definition scheme defines the *meta model* that can be tailored and *minimised* for the purposes of the domain. This was utilised in the pilot by representing only domain-dependent concept classes and relations. The definition scheme also takes care of the *semantic homogeneity* of the instance world representations by defining the possible attributes and relations for the concepts. This helped the pilot users to adopt the modelling approach; they know what they can expect to find from the instance world.

The definition scheme also gives an overview of the domain serving an important part of the *common knowledge transfer* together with the with the ability to browse the instance world. The knowledge of the model was found very interesting by the pilot users that merely extracted information from the model during the first model consultations and spend less time on reading the documentation. It seems that a model as such answers many of the users' questions and even replaces some parts of the documentation. In fact, the parts of documentation describing the relations of the concepts of the model in form of tables and textual descriptions were left of from the pilot as unnecessary ones! In general, the KR and text have somewhat complementary roles in describing different dimensions of the system (as described in page 59), whereas at least in the pilot the lack of KR has earlier forced technical documentation to transfer all the knowledge alone.

In D&T models the identity of the concept instances is based on their concept class, their relations to other concepts, together with their attribute values. This gives the representations *context based identity* and some *text independence*. The object-oriented approach adopted to D&T models was found useful in the pilot, for example, when concepts of two different concept classes using numbers as identifiers (e.g., message numbers) can be separated based on the class of the concept together with the attribute information. I.e., manipulation of concepts denoted by synonym terms or identifiers is easier with object oriented approach than with, for example, controlled vocabulary approaches.

The pilot implementation of the D&T models supports graphical interfaces for *navigating* the models and *searching* based on attribute values. According to the pilot users several hours may be spent in operational use in locating a missing piece of information due to an unknown name that can be located by navigating the models. This supports our assumption that explicit representation of central domain concepts and their relations enable the users to find information that cannot be found within a reasonable time without a knowledge representation supporting navigation. It seems that this feature is extremely

valuable when needed, as often in domains with thousands of domain-specific terms, such as the pilot domain. How often this takes place is still an open issue. Searching using relations was not found to be practical for graphical interfaces in the pilot and it is now used only through the LISP interface by the text references. During model navigation the user is able to *filter* out representations based on the properties of the relations and the concepts. In the pilot the most useful filters were the classes of the concepts and the names of the relations whereas the attribute values or relation types seem to have less practical value. In the pilot the attribute values are used mainly in searching for concept instances and the use of the relation types depends mainly on the chosen task modelling approach.

Size effectiveness of the D&T models is rather good as the models and the text references occupy altogether about tenth of the volume of documents (or of the volume of inverted index files). The models can be constructed *incrementally* by building an initial model and adding new concept and relation classes to the definition scheme together with the concept instances and relations in the instance world. The major constraint is that the user should be able to rely that after adding something to the definition scheme all relevant entities are added also to the instance world. I.e., in each step the users are aware of the contents of the model based on the status of the definition scheme. The incremental modifications to text references are typically hidden from the retrieval users and may take place at any time. Model creation without using the artifact warrant principle shares characteristics of the other domain modelling approaches (cf. pp. 73-). The most distinguishing feature is the object-oriented (OO) nature of the models.[74] It emphasises definition of attributes and relations of the concepts. Thus the *efficiency of model creation* is dependent on the efficiency of OO modelling in general. The model creation efficiency with the artifact warrant principle is described later on.

One of the major advantages of the D&T models is that it naturally supports use of *mapping method combinations* by changing the point of view. Text references and text reference classes in the model access and utilise various text retrieval techniques instead of a (set of) text retrieval techniques and systems accessing a model. Furthermore, the users can choose from several text references created for each concept. For example, one reference can include only the definition of the concept whereas another contains all the texts where the concept is mentioned (as described earlier). The first advantage of mapping method combinations is that the knowledge about the most applicable technique for each concept (or a concept class) is attached to the concept, where it is easier to maintain than, for example, external tables accessed by various TR techniques co-ordinated by some external mechanism. The second advantage is the ability to use more information to the selection and application of the TR techniques than only the (name of the) concept -- the methods operating from the model can more easily utilise all the available information in a

[74] Although the D&T models are not fully object-oriented rather than object-based.

model including the attribute values, the values of related concepts etc. The third advantage is the ease of defining many different kinds of text references for the concept from the model instead of defining them externally. The fourth advantage is the ease of co-ordinating the combination of various techniques from the model instead of coordination of dataflows between the TR techniques / systems. This would, in practice, lead to building an external controller for the coordination. Altogether, when several TR techniques can be used the knowledge how they can best be applied for each concept has to be stored somewhere and has to be updated when the model is updated. The most natural place for this knowledge and the centralised control functionality needed for applying the techniques are the concepts and the domain modelling system, respectively. This is one of the major justifications in the KR oriented point of view to TR.

The two first requirements special for technical (LPS) domains are related with text retrieval process, namely problem-orientation and retrieval resource minimisation. The view to support *problem-oriented* text retrieval adopted with D&T models focuses on mapping the users information needs with domains conceptual structures. Thus the intention is to aid in locating the problem at conceptual level as soon as possible and retrieve documentation containing potentially the required information. The use of task views was ignored to a large extent in the pilot which prevented us from gaining reliable results from evaluating the impact of this aspect to the efficiency of problem-oriented text retrieval. In the pilot implementation the text retrieval was performed in a PC platform whereas the text processing, NLP, modelling, and creation of text references was performed in a UNIX workstation. With this respect the *retrieval resource minimisation* utilised the possibility to use separate processing and retrieval environments with the DTM modelling system. In the pilot this requirement is critical to the success of large scale application of the approach.

9.4.4 DTM and Requirements for Modelling

The DTM approach and the modelling system were designed and implemented keeping in mind the requirements for technical domain modelling derived from the artifact warrant principle. The experiences from the pilot display that this was worth while. In fact, these issues should be paid even more attention in the pilot. Especially closer integration of the modelling with other information systems would be useful for enhancing the version and configuration management capabilities required from a domain modelling system in some technical domains.

Database reuse for modelling in DTM system is provided by the design information input in a batch process. The object orientation of the DTM datamodel makes is flexible enabling us to import structured information from various databases and other sources of information. The actual interaction with the DTM system consumes only a small minority of the time needed for modelling, especially, if the technical overhead in creating the connections is performed only once and the use of the system is continuous. In cases, such

as the pilot, the total effort needed for model creation can be reduced by one to two orders of magnitude. A model containing tens of thousands of concept instances, relations, and text references was created within a couple of weeks of effective working time. This task would take several man-months with any comparable tool. Thus this requirement was found to be an extremely relevant requirement in the pilot domain, but even closer co-operation between the systems would be useful (see future work starting from page 146).

Database reuse for term mappings and *database reuse for structure mappings*, were both relevant in the pilot domain. The use of acronyms, identifiers, and other terminology conventions seems to be powerful in controlled domains, such as the pilot. Of the structure oriented methods the use of documentation structures and conventions was more useful than the use of information tagged in the documents -- as opposite to our expectations. The result of text reference creation process depends on two factors: how effective are the mapping methods used (effectiveness) and are we able to generalise the text references of individual concepts to text reference classes of concept classes (efficiency). Within the DTM system we are able to utilise those mapping methods that are most suitable for each case,[75] and the effectiveness is up to the methods used, not up to the models. The methods created for individual concepts can typically be generalised for class of concepts since all the information used for creation of the individual text references is (or should be) available in the model; attributes, related concepts, etc. The most problematic cases occur if the effectiveness of the mapping methods is poor and the text references have to be (partially) hand-crafted for each concept.

In our experiments the domain expert's ability to combine structural and content oriented methods enabled us to gain sufficient effectiveness with reasonable effort. The effectiveness of the simple methods in the pilot may be a inherent property of limited technical domains where the text retrieval problems seem to be somewhat simpler than in an unlimited domain of discourse. As well they may be based on the special features of the pilot. The current tests have been performed mainly with English text and the best mapping methods for Finnish may be different.

Another rather natural and encouraging possibility is that the simple, well defined concepts of the models describing technical systems are easier to map against the texts than unstructured and complicated information needs represented by the user as a query. This suggests that the two-phase retrieval via explicit knowledge representations has also other advantages with respect to text representation level approaches than unknown terminology and other language related problems. The facility to divide the TR process into user-controlled conceptual navigation and a somewhat simplified matching of concepts against

[75] In multi-lingual documentation we are also able to create the text references to the texts written in language with which the effectivenes of the available text-retrieval technigues is best. These references can be copied to the documentation written in other languages based on structural similarity of the documentations.

texts may lead to better results by dividing the process between the user and the system. The users perform the associative, context sensitive, and fuzzy matching of their information needs utilising their world knowledge and "inherent user modelling" capabilities. The existing TR techniques are then given a simplified task that suits to the capabilities of computer systems, to match a (set of) individual, well defined concept(s) against the texts.

Version development and configuration control may sound rather odd features for some TR domain modelling approaches, such as subject headings or thesauri. In the pilot the need for these requirements was evident. The support provided by the DTM system is based on the work-flow allowing encapsulation of processing intensive parts of processing from the final integration of a TRS application. This approach was found feasible for the purposes of the pilot. Still the version development and configuration management aspect is more critical and problematic than initially expected. (cf. Artifact Warrant Principle, pp. 73-)

9.4.5 Scalability and Limits of DTM

In applications where only a few casual readers access a relatively small body of documentation, the applicability of the artifact warrant principle (AWP) based DTM approach is questionable. If the ability to find a relevant text within a limited period of time is not critical to the readers, the investment of human and material resources in incorporating the modelling to the TRS is not profitable. This issue was exemplified with the case 2, the IGE library. There the DTM approach was found useful, but there was not any driving demand for implementing text retrieval systems of any kind.

The other essential part in need versus cost analysis is the cost of implementing a DTM based system. Although the automation provided by DTM approach reduces the implementation effort significantly, there may still be costs that overwhelm the benefits. The interfacing needed for extraction of information from heterogeneous databases may consume too much resources due to purely pragmatic technical aspects or problems in integrating multidatabase systems inherent in the artifact warrant principle. In the two small cases (IGE and Planner) we found out that the effort needed for integrating a DTM based retrieval system to an end user environment can also be a major factor in effort consumption.

DTM approach applied to a documentation of a couple of hundreds pages does typically not pay of. On one hand, the effort needed for hand crafting parts of the model and for integrating the retrieval environment tend to be too large. On the other, (semi-)automated modelling and rather good applicability of text reference classes indicate that the DTM approach could be applicable to somewhat larger cases, e.g., domains with round one thousand pages of documentation. This statement has significantly more reliability in cases, where automated modelling enables easy creation of versions of a TRS for different versions and configurations of documentation. In these cases the re-use gained by automated modelling is significant in a comparison with manual work.

Modelling with DTM is mostly a knowledge engineering task with its inherent problems [238], such as the possibility to find a set of domain experts to disagree on the knowledge to be fed in to the system. Due to human factors and/or vague conceptual structure of the domains (e.g., some disciplines) the experts may continuously disagree and this knowledge engineering task can become extremely complicated preventing the system to get ready in due time. In limited domains and "closed formal worlds" - as within a programming language in the IGE case - this kinds of problems are less likely to appear. We also found out that formal task modelling has not been applied widely. This means that the automation gained by the DTM approach is typically limited to domain modelling whereas representations for tasks must be hand-crafted.

Both the absence of databases that could be utilised for automated model creation and the lack of adequate text retrieval techniques for mapping the model concepts with the texts increase dramatically the modelling effort. We should remember that the use of text reference classes does not depend on availability of external databases -- they can be used with or without imported information -- but they suppose a (set of) sufficient mapping methods to exist. Based on the pilot and the two other cases we can assume the existing, even primitive, methods to have sufficient effectiveness in controlled technical (LPS) domains. In other -- less limited, less formal, and less modelled -- domains (e.g., literature) this assumption is unlikely to be valid.

Altogether, the DTM methodology is on its best in large corporate and product documentations where the automation methods and the facility to browse domain structures can be fully utilised. It is not applicable to small, non-computerised applications where the costs of incorporating the system are not justified by the benefits -- in worst cases the incorporation of yet-another-subsystem to a TRS may require much more effort than it can be expected to save.

Apart from the AWP related issues there are some rather detailed problems in the structure of the D&T models. The pilot implementation supports only text references from the individual concepts. In some cases the users may want to find texts discussing the relation of two or more concepts. This could be realised by enabling also the relations to have text references. In addition, the current D&T models use binary relations between concepts, i.e., no relations between three or more concepts can be represented as single relation entities. For example, a business transaction might be described by a relation connecting the seller, the buyer, and the object. The current possibilities are to represent these relations as separate concepts with relations to each of the related concepts (recall database modelling), or as a intersection of the sets of texts related to each of the concepts. As task modelling needs further method development it is also likely to require additional tool support.

9.4.6 KR-Oriented View to Text retrieval

The D&T models are primarily used for TR purposes and secondary for transferring domain knowledge as such. The border line of these two activities is somewhat fuzzy, as can be observed from the following list of functionalities and features. In this list some items can be characterised as features of a TRS whereas others imply that the focus has moved from TR methodologies using domain descriptions to domain models using TR techniques.

- Supporting domain model and text retrieval systems creation process.
- Transferring an overview of the conceptual structure of a domain.
- Interchangeability and parallel use of various text retrieval methods.
- Transferring and replicating retrieval knowledge encoded in text references.
- Integration of TR domain models with other information systems by information interchange.
- Finding terms and locating concepts without a priori knowledge of their name is possible based on their relations to other concepts.
- Organising and controlling the use of retrieval knowledge according to the concepts and concept classes.
- Fulfilling some users' information needs by the models, without documentation.

It should be noticed that the representations used in the D&T models were designed primarily to support TR and only secondary to transfer knowledge as such. The majority of the information content in the model is declarative in order to maintain good interchangeability of information between various systems. The procedural knowledge encoded in the text references is minimised and encapsulated. This has reduced the hand-crafting needed for modelling.

There are also applications, where the stereotypical patterns of the conceptual structures can be utilised by other computational procedures. For example, task sequences for diagnostics, computer aided learning, etc. may be generated by rather limited, procedural or logical operations associated with each concept class or concept (cf. DMG in p. 52.). This kind of "computer aided conceptual text retrieval engineering" applications build on top of text databases are rather sensitive to the changes of the definition scheme and should thus be programmed with very flexible tools.

9.4.7 DTM and Other Systems

When the D&T models are compared with thesaurus systems, it can be stated that the D&T models can be used for emulating thesauri defining a concept class *Thesaurus Term* that has one attribute *term,* and relations to other terms, i.e., *broader than*, *related terms*, etc. Figure 25 represents one possible definition scheme for this purpose. There each term is represented as a concept associated with a *name*, a list of *used for* terms, and relations to

other terms presented as concepts at the KR level. There are also a couple of text reference classes representing stereotypical content-based text access methods used with thesauri.

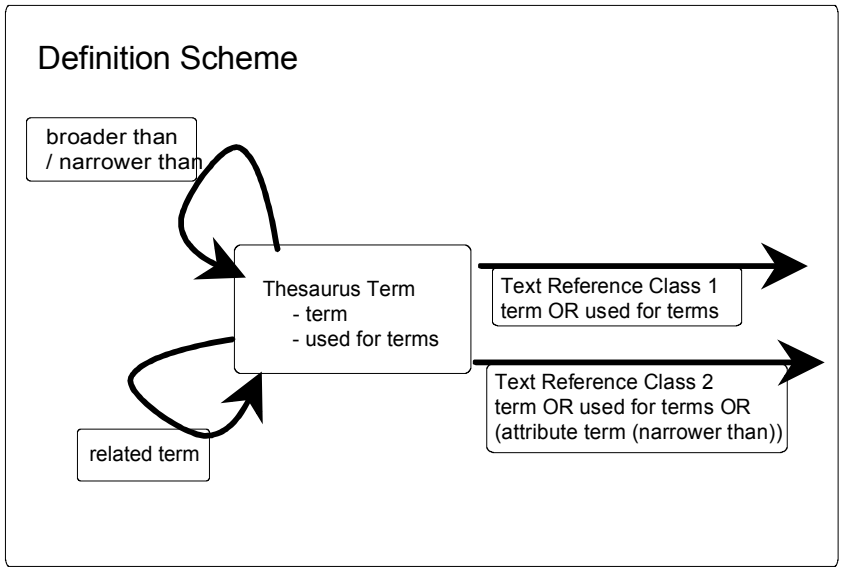

Figure 25. DTM definition scheme simulating a thesaurus.

A definition scheme emulating a subject classification system is even simpler than the one emulating a simple thesaurus. The simplicity makes both of these approaches relatively easy to adopt, but leaves the semantics very general and vague. Instead, many of the advanced thesaurus systems prove a variety of relations, but they are typically predefined and thus do not support domain-specific meta models, such as the definition schemes in D&T models. They do also use terms instead of objects to represent the concepts elaborating the use of various concepts for "synonym" concepts and forcing use of artificially created multi-word terms for representation of attribute information, f.ex., "four piston engine with 2.0 litter volume with two twin cams" as a synonym of "mx204tc26gh", a product code. Alternatively, the attribute information could be represented as first level objects linked to the primary objects (i.e., "term concepts") with relations named according to the attribute names. This kinds of data structures would be very simple at the lowest level, but the management of classes, attribute and relation specifications etc. with the same scheme would be less tempting. Treating attributes as first level objects is also not preferred from the knowledge representation point of view. Consider, for example, attribute values whose semantics entirely depend on the context of the object, such as "version 1.2.1". In general, the basic thesaurus systems are more suited to less structured, more vague domains and for text representation level (i.e., term) operations.

Similarly, the D&T models can be used for emulating basic the hypertext systems by creating a concept (instance of a concept class *Concept-defined-by-a-node*) for each node of the hypertext. From the hypertext point of view the D&T models can be seen as a hypertext network that is divided into three parts: the two-level semantic network, and the information modules; texts, graphics, etc. From the library science point of view the D&T models can

bee seen as an augemented thesaurus linked with other functionalities of a text retrieval system. From NLP point of view the D&T models are interlinked knowledge representations selected from the NLU world knowledge. All these statements can be supported to some extent.

The major differences between other approaches and the DTM modelling approach used with the D&T models can be encountered by comparing the other approaches referred to in this work with the requirements stated in pages 81-85. The approaches listed below are selected examples from alternative points of view; from hypertext oriented advanced thesaurus approaches, from IR oriented KR, from hypertext oriented NLP, and from IR DB integration. All these systems have many features similar to the D&T models, but they have some problems, such as meta model minimisation and semantic homogeneity (i.e., requirements 3 and 4, respectively). The major problems of these approaches are in the support for automated modelling. They do not satisfy the creation efficiency and combination flexibility requirements (14 and 15) and not especially the requirements related with the artifact warrant principle (III-VI). Anyhow, in most of these approaches the incorporation of the AWP would be possible and most useful for their applicability to technical LPS documentation domains. For further references see the beginning of this work.

RADA [214] is a somewhat similar approach from the thesaurus point of view. It is a graphical thesaurus like approach, where concepts and their relations can be represented. The identity of the concepts is based on their names and their position in the network, i.e., they have no attributes. RADA does not support efficient modelling; the mappings to texts are based on keyword lists assigned for each concept, and the models are created manually. There exists also other approaches separating the hypertext nodes and a conceptual level represented by interrelated "concept terms" [3, 145]. The concepts may also be classified, but typically lack structure, semantic homogeneity, or means needed for utilisation of design information. For other thesaurus oriented hypertext approaches (like Rada's [177]) see the early chapter representing the various approaches.

ARGON [167] utilises knowledge representation for information retrieval. The KR contains frame taxonomies with non-hierarchical relations, attributes as well as browsing facilities. Thus the internal structure of the KR model is close to the D&T models. In ARGON the information retrieval facilities are limited to the retrieval of model entities and no attention has been paid to the mappings with texts in text databases, i.e., utilisation of various text processing and retrieval facilities to link texts with the model. Modelling in Argon is manual, but could utilise artifact warrant principle.

TOPIC [79] provides text parsing and generation of interlinked concept frames from text. HYPER-TOPIC [123] built on it enables generation of hypertext based on the extracted concept frames. This approach to modelling includes also statistical features producing, but could be enhanced with use of design information. Also the pragmatic issues

(processing of large documentations, utilisation of document structures and design information, efficient support form browsing, etc.) seem to need some further work in this system.

The integration of TRS with other information systems is discussed in the context of intelligent databases (see page 54). One of the approaches containing elements relevant to the artifact warrant principle is Soergel [204] discussing integration of database and retrieval systems. In his example, he integrates database describing university entities, such as teachers, students, books, and courses with subject classifications. Unfortunately the integration is limited to a subject field added as an attribute to other entities. Thus there is no direct connection from the database entities to the texts, i.e., only the route via the subject field is available and the database information is not used for TR modelling. Similarly Agosti et. al. [2] focuses more on modelling a TRS rather than the domain it operates in.

10 SUMMARY

This work addressed the problems of transferring knowledge of large technical products and systems by combining natural language and formal knowledge representations (KR). The approach adopted here reduces the problems in combining KR and text mainly to KR supported text retrieval. The focus is on domain modelling techniques to be incorporated to technical design and production processes.

The first contribution of this work is a framework for representations of text retrieval systems used for comparison of different approaches and analysis of their problems. The analysis adopts a view where the systems' representations (2-4) between the users' mental images of their information needs (5) and the texts satisfying these needs (1) are grouped as syntactically extracted text representations (2), vocabulary independent knowledge representations (4), and mappings (3) implementing relations between these two. The point of view and level of abstraction chosen in the framework were sufficient for characterising the inherent features and analysing the problems of various approaches from a common basis. The thesis presented also the DMG system utilising explicit separation of the representation levels for generation of hypertext advisory systems from a knowledge representation.

This work focused on technical documentation. Its' role was seen as a paper-based media for transferring knowledge needed for solving problems needed to solve for accomplishing the mission of an organisation. The close relation of highly structured technical documentation, product knowledge, and electronic design databases was described. The problem-oriented nature of technical documentation retrieval was described with effort/result diagrams. The total economical impact of a text retrieval system requires comparison of the system's affect to the organisation's success criteria and efficiency with the costs of the system, although these are difficult to measure as such. The cost considerations imply that the invisible text retrieval costs can often be reduced by investing on preparations, such as domain modelling. Still, maintenance of accurate retrieval structures must be achieved with highly automated means.

The analysis was next focused on domain modelling in technical domains. Technical documentation describing large products and systems as well as it's creation and use were analysed. The special features of technical domains imply special constraints and opportunities for construction of domain models supplementing text retrieval. The work contributed by presenting artifact warrant principle (AWP) recommending the use of design information in creation of domain models for technical domains. The constraints and opportunities denoted by the AWP were not found to be in accordance with contemporary general purpose modelling methods. Specialised modelling methods, possibly supported by specialised representations, can be devised for these purposes. On the basis of this analysis

and previous parts of this work the thesis contributed by describing requirements for knowledge representations used for KR supported technical documentation retrieval.

The thesis described D&T (domain and task) models designed on the basis of the requirements presented for KR. DTM modelling system is a domain independent environment for creation of D&T models developed jointly by the author and other members of the research group at NRC as a part of the project SIMPR. The D&T models emphasise explicit representation of central domain structures and adopt a domain-centered view to text retrieval as opposite to the approaches concentrating on the text and text representations. The D&T models apply two-phase text retrieval approach derived from the level for TR representations. Matching users' information needs with the conceptual representations at the KR level is controlled and performed by the users. The simplified text retrieval problem to retrieve documentation about each single well-defined concept is performed by the system (according to the domain-dependent TR knowledge encoded in text references).

The next contribution is the design of the DTM modelling approach derived from the KR requirements the artifact warrant principle. DTM methods enable efficient and economical application of the D&T models in technical domains. The description starts by a quantitative analysis of the models, a comparison with hypertext creation, and presentation of the CIOS architecture (concept Classes, concept Instances, text Objects, and text Segments) reducing the magnitude of links in hypertext systems. The artifact warrant principle is supported by the modelling methods based on the two-level class-instance structure of the D&T models. These are semi-automated model creation based on design information input and semi-automated creation of text references using text reference classes. The chapter finished by guidelines for DTM domain modelling.

Two small cases and a large pilot text retrieval system called TeleSIMPR were described validating the artifact warrant principle, the requirements for text retrieval KR, the D&T models, and the DTM methods in an industrial application. In the pilot a D&T model was used mainly to aid in text retrieval, but also as such to transmit parts of the domain knowledge to the users in form of a KR. The D&T models were preferred for transferring very structured, homogeneous knowledge. Also some vocabulary-dependent problems of contemporary TR approaches were avoided by KR supported TR. In the pilot the knowledge engineering effort needed for hand-crafting the domain models was reduced by utilising existing information systems according to guidelines derived from the artifact warrant principle presented earlier. Similarly, the DTM methods automated the creation of mappings from the conceptual representations to the texts using available text retrieval methods. As a result, the first commercial technical documentation retrieval system utilising the DTM techniques will be released in the near future.

11 CONCLUSIONS

Based on the analysis of existing text retrieval approaches and technical environments this work indicates that technical documentation describing large products and systems as well as its' creation and use are inherently differ from non-technical documentation. The documentation is closely related with the design databases and somewhat oriented towards formal knowledge representations. Text retrieval is characterised by problem-orientation and vast amounts of domain-specific concepts. These imply special constraints and opportunities for construction of domain models supplementing text retrieval, that are not in accordance with contemporary general purpose modelling methods. Still, specialised modelling methods -- possibly supported by specialised representations -- can be devised for these purposes leading to improved modelling performance and advantages in technical knowledge transfer.

The approach of combining knowledge representations (KR) and text as KR supported text retrieval (TR) adopted in this work resulted in an independent domain modelling component used with interchangeable TR techniques. Based on the DTM domain modelling system demonstrating these ideas and the results from a pilot this approach were found to be feasible for large product and system documentations. It can be stated that in modelling technical domains, such as the pilot, the ability to find adequate abstractions enabling similar semantic treatment of groups of concepts enables automation and also reduces users cognitive overload during text retrieval. Formal KR, such as the D&T models, are preferred for transferring very structured, homogeneous knowledge. A good basis for automation exists if the domain experts are able to agree on a homogeneous conceptual structure of a domain, data from existing (design) information systems can be utilised, and the text retrieval techniques available are sufficient for matching the single concepts of the domain against the texts. On the basis of the analysis and the positive experiences from the pilot this thesis states that the advantages of automation are best achieved by integrating the domain modelling for text retrieval purposes closely with other information systems.

12 FUTURE RESEARCH

12.1 Extensions to D&T Models

The pilot case showed the domain modelling methods developed to have significant practical value. There are still many issues that need some further consideration, such as the optimal task modelling approaches and the optimal size of the models with respect to their coverage of the domain. Following is a list of issues we would like to investigate in the near future:

Model Consistency Checking and Quality Analysis

The quality and consistency of the model and text reference creation processes require some further analysis. Tools are needed for analysing statistical features of the models, such as distribution of texts per text reference. Especially we need to analyse the quality of the mapping creation (precision/recall) in order to compare the generic methods with methods utilising domain-specific conventions, such as use of pre-defined fields of documents. If the effectiveness of these mapping methods degrades, the productivity of the text reference class technique decreases, since the automatically created text references have to be validated manually. We intent also investigate under which conditions the information available in the model is sufficient for creation of effective text reference classes and thus sufficient for efficient mapping creation.

Extensions to Use of Text References

The current DTM methods support creation of text references for concept instances and concept classes. In some cases the users wish to retrieve the texts describing the relation between two concepts. This would be possible by enabling use of text references for relations. In addition, the exists a variety of techniques for combining, weighting, and ranking texts referred to by a pair or a set of interesting concepts that may in some cases be useful also in the DTM system.

Text Object Classes

Currently the D&T models used in the pilot system treat the texts as uniform text objects of a single text object class. The full exploitation of document structures would be easier if the texts were separated into several classes with specific structure and semantic expectations. This can be utilised especially for separating different texts into different views. Consider, for example, definition of an "overview" view to the text references by including only those text objects that are instances of the *overview-chapter* class. After this extension the manipulation of the model and manipulation of the texts would be more alike -- there would

exist text object classes and text objects analogously to the concept classes and concept instances. This would also enable us to treat the text references linking concept instances with text objects somewhat analogously to the relations linking concept instances with other concept instances, manipulate text reference classes somewhat analogously to relation classes, etc.[76] Currently the differentiation between text objects is based on the master heading hierarchy of the documentation, i.e., on the deviation of the texts to different document databases.

Scaling up the Models

The current architecture is designed for a text database containing about a million text objects. When the number of texts increases, the average number of relations from a concept instance to texts increases respectively, until becomes unmanageable. The means to cope with this problem include:

- reducing the number of unnecessary text references, i.e., improving precision by utilising combinations of alternative methods, especially text object classes or other structural methods in a combination with content oriented methods,
- reducing the number of text references to composite text objects by replacing a set of references to sub-chapters with one reference to the main chapter,
- dividing the concepts set of text references into smaller, classified subsets using,
  - view types in a combination with the selective mapping methods and text object classes, or
  - index phrase collections for each concept instance,
- increasing the number of the concept instances, and
- increasing the number of the concept classes.

The incorporation of these ideas are rather straight-forward in the DTM environment. The reliable evaluation of these ideas would require a documentation of this size in real-life use in order to gain insight on the real problems and relevant criteria for the evaluation.

12.2 Dynamic Models

In some cases the information, on which the model should be based on, changes so rapidly, that a batch conversion from external databases to a model is not sufficient. Situations like this can be managed by a dynamic model. This means that the instance world of the model does not exist explicitly, but is defined only implicitly, as methods of the classes in the definition scheme. Each of the concept classes is associated with methods to access the attributes and the relations (or related concept instances) of a concept instance based on the identifier of an instance. Also all the text references must be based on text reference classes

[76] This extension is invisible for the users retrieving text using the models, i.e., has no influence on the CIOS retrieval architecture.

of the concept classes, which is possible only if the texts related to all concept instances of a concept class can be found with a set of general methods.

Example. Let us suppose, that we have a relational database, where the information about the parts of cars is stored as relational tables. The text retrieval starts, as usual, from the definition scheme. When the user wants to select one concept from the class *car type*, selected fields of the table "type-tbl" are displayed to the user. The *motor* related to the selected *car type* can be found using a query to another relation table based on the identification of the selected *car type*. Further, the attribute values of the *motor* may be collected from two other tables. At last, the texts describing the selected *motor* are found using a query of the *product code* column of the motor table, or alternatively, from the document whose name is in another column of the same table.

The use of dynamic models does not require some specific representation, such as the relation tables above. In some cases a computational representation is preferred to an explicit representation. Consider, for example, method to encode the parts of a product in a product code so that the enumeration of all possible feature combinations would take huge amount of space whereas the product codes can easily be encoded from the product code to the sub-part list of the product. In general, the requirements for use of these dynamic models are that (1) the identity of each concept can be maintained during the retrieval process, (2) the data (database tables, text files, etc.) is available via computer networks, (3) the computations needed for the process (queries, production of part lists based on product feature codes, etc.) are fast enough to satisfy the user needs, and (4) the text retrieval methods defined for the concept classes are adequate for all concept instances of each class. Unfortunately these requirements are typically not fulfilled due to lack of some information, fragility of networks, or at least due to too long response times. Even though, we aim at evaluating the usefulness of the dynamic models and in developing techniques to overcome the obstacles described above.

12.3 E/R Diagrams and Performance Evaluation

There should be metrics for evaluating the TRSs from the point of view of effectiveness, efficiency, and costs of the problem-oriented text retrieval process discussed above. Unfortunately the most commonly used measures used for evaluation of TRS performance, precision and recall, are not adequate for evaluation or comparison of the developed methods. The problems are mainly based on the tradition of query-based information retrieval systems and on the fact that the P/R values are calculated based on the evaluation of a single query in order to determine the effectiveness of the retrieval algorithm and not the effectiveness or efficiency of the whole TR process. The main problems considered here are (for further details see [229]:

1. Determination of the set of relevant texts is elaborate and inaccurate.[77]
2. User competence to formulate queries affects the results.[78]
3. Effort needed all parts of the TR process should be measured.
4. Sum effect of more than one queries should be measured.
5. Applicability to non-query TR paradigms.
6. Environment-specific parameters.

Clearly the P/R measures are not adequate for the measures of the whole TR process. A simple, very general alternative is to measure user preferences between evaluated systems, but there is no practical way to arrange tests large enough to be statistically relevant. Neither do the other contemporary measures,[79] fit in the needs of problem-oriented text retrieval. The use of the effort/result diagrams (cf. p. 66) as the metrics might offer some advantages in comparison with the P/R measures (compare with the previous list):

1. The determination of the set of relevant texts is easier. If the result to the problem was found, no further relevance determination is needed. If it was not found, only the existence of any text containing the answer is needed.
2. User competence has still effect on the results, but these measures do show better values for systems that enable casual users to use the systems.
3. The effort needed for the whole TR process is measured.
4. Sum effect of more than one queries is measured.
5. The method is applicable also for non-query-based methods such as hypertext systems. The values measured in same environments are comparable.

Of the problems listed in the previous page the effort/result consideration gives no relief to the problem of comparing two measures where the environment or the documentation is not equal. This problem is inherent for measures of the whole TR process and can be avoided only by using a wide selection of standardised test environments. Ease of determining the relevant texts in effort/result tests enables use of larger collections, although the human work is not eliminated (in fact, the evaluation is still elaborate). The restriction to *successful/not found/incorrect stop* results is thus on one hand enabling the use of large test collections but on the other limiting the information gained from each process. The percentage of found relevant texts is a still more useful measure in environments, where all texts containing the query concept should be found. Also the systems ability to aid users is

---

[77] See for example [63, 64]. Further, in some cases only one text defining the concept contained in a query is considered relevant whereas in some cases all occurrences of the concept should be retrieved. The statistical methods seem to be more oriented towards the first goal.

[78] In general, user competence to formulate queries is measured instead of user-friendliness of the system that is more relevant for the whole process. The relevant statements for evaluation are like "46% of searches by end-users were complete failures" or "45% of searches retrieved no documents" instead of "on an average, 45% of relevant texts were retrieved".

[79] Such as usefulness [63], precision-document [133], informativeness-time-user friendliness [213], or cost considerations in [34]. See also [15, 73].

measured by the effort/result diagrams, but the competence of users has still affect on the measures.

Our intention is to develop further this preliminary idea of using effort/result diagrams derived from users' performance in doing controlled tasks to comparison of different text retrieval approaches. We have to develop also some supplementary techniques to decrease the effort needed from the subjects and for the test set-up. One practical approach is to embed some extra functionalities to the retrieval interface aiding the tests, for example, by writing a log (trace file) of the user interaction. This could be used for recording each problem, the steps of the processes, and the results of the processes needed for quantitative and qualitative analysis. If these two tasks were completed successfully, the basis for a successful and comprehensive evaluation were build (cf. p. 130), and we were able to evaluate quantitatively the retrieval efficiency with D&T models and alternative methods in the pilot environment.

12.4 Task Sequence Modelling

The trace files collected for the retrieval process evaluation (as sketched above) would describe also the paths the users navigate trough a model as they perform a task. The log facility and the data collected could also be utilised for storing the task sequences performed for further use and for analysis of the user behaviour during the retrieval process. When a path navigated through once is stored as a saved session containing a path or trail of navigated concepts and texts, the user can use it as a guideline for performing the task next time. If enough paths navigated by the users would bee collected (by storing the sessions or by simply collecting all log files), we were able apply case-based reasoning (CBR) techniques for aiding the future users in performing the tasks. Further, we might be able to abstract generic task descriptions on the level of concept classes.

Consider, for example, the task of changing oil to an engine (cf. the example in p. 90). If the user has located the concept instance of *engine* class from the model, (s)he is likely to consult the texts in the *maintenance* view of the *oil* instance the *engine* instance *uses*. This kind of stereotyped navigation patterns containing long sequences of navigation can probably be abstracted at the level of the concept classes and view types. In addition to the previous cases, we are able to compare them with the task models explicitly created to the model and thus to suggest next step in the navigation process according to a recognised beginning of the sequence.

This idea and the use of hypertext networks for task guidance can be compared from the point of view of the CIOS retrieval architecture. Hypertext networks displaying the texts needed for performing a task are encoded as paths or trails of nodes at the text object level. The CBR reasoning techniques utilise the previous paths of concept instances (and text objects connected to them) at the concept instance level. If we are able to abstract the

generic tasks at the level of concept classes, we are two levels and some orders of magnitude above the manual encoding of hypertext advisory systems.

REFERENCES

1. Adelson, B., Comparing Natural and Abstract Categories: A Case Study from Computer Science, Cognitive Science (1985), No. 9, pp. 417-430.
2. Agosti, M. Crestani, F. Gradenigo, G., Towards data modelling in information retrieval, Journal of Information Science (1989), No. 15, pp. 307-319.
3. Agosti, M. Gradenigo, G. Marchetti, P.G., A Hypertext Environment for Interacting with Large Textual Databases, Information Processing & Management, Vol. 28 (1992), No. 3, pp. 371-387.
4. Alkula, R. Honkela T., Kielianalyysiohjelmien soveltaminen tekstitietokannoissa - FULLTEXT-projektin tuloksia, (in Finnish, title in English: Application of Language Analysis Programs in Text Databases - Some Results from the FULLTEXT Project), Hand-out material and verbal information at Tutkittua tietoa tekstitietokannoista seminar, VTT/Info, Vuorimiehentie 5, FIN-02140 Espoo, Finland, May 11, 1992.
5. Allen, J., Natural Language Understanding, Benjamin/Cummings Publishing Company, 1988.
6. Allen, T.J., Managing the Flow of Technology: Technology Transfer and the Dissemination of Technological Information within the R&D Organisation, Massachusetts Institute of Technology, 1977.
7. Andersen, A. Munch, K.H., Automatic Generation of Technical Documentation, Expert Systems with Applications, Vol. 3 (1991), pp. 219-227.
8. Anick, P.G., Integrating 'Natural Language' and Boolean Query: An Application of Computational Linguistics to Full-text Information Retrieval, Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991.
9. Antoniadis, G. Lallich-Boidin, G. Polity, Y. Rouault, J., A French Text Recognition Model for Information Retrieval System, ACM SIGIR, Proceedings of 11th International Conference on Research & Development in Information Retrieval, Grenoble, France, June 13-15, 1988, pp. 67-83.
10. Appel, R.D. Komorowski, H.J. Barr, C.E. Greenes, R.A., Intelligent Focusing in Knowledge Indexing and Retrieval - The Relatedness Tool, Proceedings of the 12th Annual Symposium on Computer Applications in Medical Care, November 1988, pp. 152-157.
11. Artificial Intelligence, Special Volume on Knowledge Representation. Vol. 49, (1991), No. 1-3.
12. Barnett, J. Knight, K. Mani, I. Rich, E., Knowledge and Natural Language Processing, Communications of the ACM, Vol. 33 (1990), No. 8. pp. 50-71.
13. Barrett, E., The Society of Text, The MIT Press, 1989.
14. Bates, M.J., Information Search Tactics, Journal of the American Society of Information Science, July 1979, pp. 205-214.
15. Belkin, N.J. Oddy, R.N. Brooks, H.M., ASK for Information Retrieval. Part 1. Background and Theory, Journal of Documentation, Vol. 38 (1982), No. 2, pp. 61-71.
16. Berry, E., How to Get Users to Follow Procedures, IEEE Transactions on Professional Communication. Vol. 25 (1982), No. 1, pp. 22-25.
17. Betts, R. Marrable, D., Free text vs controlled vocabulary - retrieval precision and recall over large databases, Proceedings of Online Information 91, London, England, December 10-12, 1991. Learned Information Ltd. pp. 153-165.
18. Bieber, M.P. Kimbrough, S.O, On Generalizing the Concept of Hypertext, MIS Quarterly, March 1992, pp. 77-93.
19. Black, W.J., Knowledge-based abstracting, Online Review, Vol. 14 (1990), No. 5. pp. 327-340.
20. Blair, D.C. Maron, M.E., Full Text Information Retrieval: Further Analysis and Clarification, Information Processing and Management, Vol. 26 (1990), No. 3, pp. 437-447.
21. Bonzi, S. Liddy, E., The Use of Anaphoric Resolution for Document Description in Information Retrieval, ACM SIGIR, Proceedings of 11th International Conference on Research & Development in Information Retrieval, Grenoble, France, June 13-15, 1988, pp. 53-66.

22. Bourdage, R., Slice of Life STAKAuthor: A Second Generation Hypercard Authoring Tool, Hypermedia, Vol. 2 (1991), No. 3, pp. 249-157.

23. Brachman, R.J. Levesque, H.J. (eds.), Readings in Knowledge Representation, Morgan Kaufman. Los Altos, CA, 1985.

24. Brajanik, G. Guida, G. Tasso, C., IL-NLI II: Applying Man-Machine Interaction and Artificial Intelligence concepts to Information Retrieval, ACM SIGIR, Proceedings of 11th International Conference on Research & Development in Information Retrieval, Grenoble, France, June 13-15, 1988, pp. 387-399.

25. Brown, J., Research that Reinvents the Corporation, Harvard Business Review, January-February 1991, pp. 102-111.

26. Carando, P., SHADOW Fusing Hypertext with AI, IEEE Expert, Vol. 4 (1989), No. 4, pp. 65-78.

27. Carlson, P.A., Hypertext and Intelligent Interfaces for Text, in Barrett, E. (ed.), Society of Text, MIT, Massachusetts, 1989, pp. 59-74.

28. Carlson, D.A. Ram, S., HyperIntelligence: The Next Frontier, Communications of the ACM, Vol. 33 (1990), No. 3, pp. 311-321.

29. Carnegie Group's Text Gategorization Shell, Commercial material, Carnegie Group Inc. Pittsburgh, Pa, USA. 1989.

30. Case, D.O., How do the experts do it? The use of ethnographic methods as an aid to understanding the cognitive processing and retrieval of large bodies of text, ACM SIGIR, Proceedings of 11th International Conference on Research & Development in Information Retrieval, Grenoble, France, June 13-15, 1988, pp. 127-133.

31. Celentano, A. Fugini, M.G. Pozzi, S., Semantic Retrieval of Documents: A framework for a Knowledge-Based System, A.I.C.A. Annual Conference, Trieste, Italy, 4-6 October 1989.

32. Chomsky, N.A., Syntactic Structures, Mouton, The Hague. 1957

33. Cleverdon, C.W., Optimising Convenient On-line Access to Bibliographic Data Bases, Information Services and Use, Vol. 4 (1984), No. 1, pp. 37-47.

34. Cleverdon, C.W., The Significance of the Cranfield Tests on Index Languages, ACM SIGIR, Proceedings of 14th International Conference on Research and Development in Information Retrieval, Chicago, Illinois USA. October 13-16, 1991, pp. 3-12.

35. Cleverdon, C.W. Mills, J. Keen, E.M., Factors Determining the Performance of Indexing Systems, Vols. 1, 2 Aslib, Cranfield Research Project. Cranfield, England, 1966.

36. Collins, A. Smith, E.E. (eds.), Readings in Cognitive Science, A Perspective from Psychology and Artificial Intelligence, Morgan Kaufman Publishers, Inc. San Mateo, California, 1988.

37. Communications of the ACM, Special issue on Next-Generation Database Systems, Vol. 34 (1991), No. 10.

38. Conklin, J., Hypertext: An Introduction and Survey, IEEE Computer, (1987), No. 9, pp. 17-41.

39. Cox, B.J., Object-oriented Programming: An Evolutionary Approach, Addison-Wesley, Reading, MA, 1986.

40. Créhange, M. Foucaut, O. Halin, G. Mouaddib, N., Semantics of User Interface for Image Retrieval: Possibility Theory and Learning Techniques, Information Processing & Management, Vol. 25 (1989), No. 6, pp. 615-627.

41. Croft, W.B. Lucia, T.J. Cohen, P.R., Retrieving Documents by Plausible Interface: A Preliminary Study, ACM SIGIR, Proceedings of 11th International Conference on Research & Development in Information Retrieval, Grenoble, France, June 13-15, 1988, pp. 481-494.

42. Croft, W.B. Turtle, H.R. Lewis, D.D., The Use of Phrases and Structured Queries in Information Retrieval, ACM SIGIR, Proceedings of 14th International Conference on Research and Development in Information Retrieval, Chicago, Illinois USA. October 13-16, 1991, pp. 32-45.

43. Cutler, M., Query-based and Linked-based Information Retrieval in Full Text Databases, Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991.

44. Dahlgren, K. McDowell, J., Knowledge Representation for Commonsense Reasoning with Text, Computational Linguistics, Vol. 15 (1989), No. 53, pp. 149-170.
45. Davis, M.D. Weyuker, E.J., Computability, Complexity and Languages, Academic press Inc. New York, 1983.
46. Dengel, A. Mattos, N.M. Mitschang, B., An Integrated Document Management System, SPIE, Vol. 1293, Applications of Artificial Intelligence VIII, 1990, pp. 368-379.
47. Dervin, B. Nilan, M., Information Needs and Uses, in Williams, M.E., Annual Review of Information Science and Technology, American Society of Information Science (ASIS), Vol. 21, Knowledge Industry Publications, Inc. NY. 1986, pp. 3-33.
48. Dick, J.P. Hirst, G., Intelligent Text Retrieval, Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991.
49. Dillon, A., Readers' models of text structures: the case of academic articles, Man-Machine Studies (1991), No. 35, pp. 913-925.
50. Directory of Online Databases, Cuadra-Elsevier, New York, NY. Vol. 9 (1988), No. 1.
51. Driscoll, J.R. Rajala, D.A. Shaffer, W.H. Thomas, D.W., The Operation and Performance of an Artificially Intelligent Keywording System, Information Processing & Management, Vol. 27 (1991), No. 1, pp. 43-54.
52. Dun, Bradstreet, Observing Europe, ECHO (European Comission Host Organization) Information Market online database, ENR: 392, DG XIII of Comission of the EC, 1990.
53. Duncan, E.B., Structuring Knowledge Bases for Designers of Learning Materials, Hypermedia, Vol. 1 (1989), No. 1, pp. 20-33.
54. Duncan, E.B., A Concept-Map Thesaurus as a Knowledge-Based Hypertext Interface to a Bibliographic Database, Proceedings of Procpects for Intelligent Retrieval, Informatics 10, Cambridge, UK, 21-23 March 1989, (Aslib, UK. 1990), pp. 43-52.
55. Electronic Manuscript Preparation and Markup, American National Standard Z39.59-1988. National Information Standards Organization (NISO), USA, 1988.
56. Engelbarth, D.C., The Augemented Knowledge Workshop, Proceedings of ACM History of Personal Workstations, Paolo Alto, CA. January 9-10, 1986, pp. 73-83.
57. Evans, D.A. Handerson, S.K. Lefferts, R.G. Monarch, I.A.., A Summary of the CLARIT Project, Report No. CMU-LCL-91-2, Laboratory for Computational Linguistics, 139 Baker Hall, Carnegie Mellon University, Pitsburgh, November 1991.
58. Fagan, J., Experiments in Automatic Phrase Indexing for Document Retrieval: A Comparison of Syntactic and No-Syntactic Methods, Ph.D. Thesis, Technical Report 87-868, Cornell University, Computer Science Department, 1987. (UMI Dissertion Services 1988).
59. Fauconner, G., Domains and connections, Cognitive Linguistics, Vol. 1 (1990), No. 1, pp. 151-174.
60. Finin, T. McEntire, R. Weir, C. Silk, B., A Three-Tired Approach to Natural Language Text Retrieval, Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991.
61. Fox, E.A., Testing the applicability of intelligent methods for information retrieval, Information Services & Use, Vol. 7 (1987), No. 5-6, pp. 119-138.
62. Frappaolo, C., A New Look at an Old Idea, Document Abstracting, Text Management Journal, Delphi Consulting Group, Inc. Vol. 1 (1990), Issue 2, pp. 30-32.
63. Frei, H.P. Schäube, P., Determinating the Effectiveness of Retrieval Algorithms, Information Processing and Management, Vol. 27 (1991), No. 2/3, pp. 153-164.
64. Froehlich, T.J., Relevance and the Relevance of Social Epistemology, in Hämäläinen, P. Koskiala, S. Repo, A.J. (eds.), 44th FID Conference and Congress,. Paer I. International Federation for Information and Documentation, Finnish Society of Information Services, 1988, pp. 19-28.
65. Furnas, G.W. Landauer, T.K. Gomez, L.M. Dumais, S.T., The Vocabulary Problem in Human-System Communication, Commendations of the ACM, Vol. 30 (1987), No. 11, pp. 964-971.

66. Futrelle, R.P. Cleary, M.E., Tools for Natural Language Processing that can Aid Retrieval from Technical Text, Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991.

67. Gallant, S.I., Context Vector Representations for Document Retrieval, Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991.

68. Garg, P.K. Scacchi, W., ISHYS, Designing an Intelligent Software Hypertext System, IEEE Expert, Fall 1989, pp. 42-63.

69. Gauch, S. Smith, J.B., Search Improvement via Automatic Query Reformulation, ACM Transactions on Information Systems, Vol. 9 (1991), No. 3, pp. 249-280.

70. Gazdar, G. Klein, E. Pullum, G. Sag, I., Generalized Phrase Structure Grammar, Blackwell, Oxford, England, 1985.

71. Gelbart, D. Smith, J.C., Current issues in Text Retrieval: FLEXICON, A Legal Text-Based Intelligent System, Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991.

72. Gibb, F. Smart, G., Structured Information Management Using New Techniques for Processing Text, Online Review, Vol. 14 (1991), No. 3, pp. 159-171.

73. Goldman, S.R. Dolan, C.P., Looking over the User's Shoulder, A Pragmatic Approach to Automated Text Processing and Retrieval, Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991.

74. Gomez, F. Segami, C., Knowledge acquisition from natural language for expert systems based on classification problem-solving methods, Knowledge Acquisition, (1990), No. 2, pp. 107-128.

75. Goodman, N., Languages of Art, an Approach to a Theory of Symbols, 2nd ed. (1976), Hackett Publishing Company, Inc. USA. 1988.

76. Guidelines for the Establishment and Development of Monolingual Thesauri., ISO 2788 (TC 46, Information and Documentation). Paris. International Organization for Standardisation (ISO). 1986.

77. Güntzer, U. Jüttner, G. Seegmüller, G., TEGEN - ein lernfähiges Information Retrieval System, in English: TEGEN - a self-adaptive Information Retrieval System), Proceedings of the international congress of Terminology and Knowledge Engineering, INDEKS Verlag, Frankfurt/M, 1987. pp. 323-337.

78. Haas, S.W., Unknown Words: Is the Choice of Dictionary Important in Text Retrieval? Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991.

79. Hahn, U. Reimer, U., Knowledge-Based Text Analysis in Office Environments: The Text Condensation System TOPIC, in Lamersdorf (ed.), OFFICE KNOWLEDGE: Representation, Management, and Utilization, Elsevier (North-Holland), 1988, pp. 197-215.

80. Hamilton, R. Hamilton, D., On-Line Documentation Delivers, Datamation, July 1. 1990, pp. 45-50.

81. Harman, D., Closing the Vast Technology Transfer Gap in Text Retrieval, Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991.

82. Harmon, P. King, D., Expert Systems, John Wiley & Sons, Inc. New York, 1985.

83. Harris, C.L. et al., Office Automation: Making it Pay Off, in Forester, T. (ed.), Computers in the Human Context, Information Technology, Productivity and People, Basil Blackwell, UK. 1989, pp. 367-376.

84. Harvey, D.A., Catch the Wave of DIP, Byte, April 1991. pp. 173-182.

85. Hauptmann, A.G., From Syntax to Meaning in Natural Language Processing, Proceedings of the Ninth National Conference on Artificial Intelligence, AAAI-91, AAAI Press / The MIT Press, Vol. 1, July 14-19, 1991, pp. 125-130.

86. de Hilster, D. Meyers, A., Heuristic Skimming of Voluminous Text, Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991.

87. Hesketh, P. Barrett, T. (eds.), An Introduction fo the KADS Methodology, Report from Esprit P1098 project. STC Technology Ltd. Harlow, England. 1989.

88. Hicjey, B.A. Nielsen, B. Linguistic Analysis of Reference Transactions: If You Want an Answer, You've Got to Find a Question, Expert Systems for Libraries at SCIL '88, pp. 193.

89. Hull, R. King, R., Semantic Database Modeling: Survey, Applications, and Research Issues, ACM Computing Surveys, Vol. 19 (1987), No. 3, pp. 201-260.

90. Humphrey, S.M., A Knowledge-Based Expert System for Computer-Assisted Indexing, IEEE Expert, Fall 1989. pp. 25-38.

91. Hätönen, K., Tekstitiedon hakua tukevan tietämyksen esittäminen, (in Finnish), title in English: Knowledge Representations for Text retrieval, M.Sc. thesis (Computer science), University of Helsinki, 1992.

92. Hätönen, K. Parpola, P. Rämö, K. Tyrväinen, P., On the Structures of Technical Documentation, SIMPR document No. SIMPR-NRC-1991-26-22e, September 1991.

93. Hätönen, K. Parpola, P. Tyrväinen, P., Semi-Automatic Creation of Domain Models Based on Design Information Input, SIMPR document No. SIMPR-NRC-1991-24-17e, September 1991.

94. Hätönen, K. Parpola, P. Tyrväinen, P., Text References: Methods to Link Models with Texts, SIMPR document No. SIMPR-NRC-1991-24-18e, September 1991.

95. Hätönen, K. Saarinen, P. Stickler, P. Tyrväinen, P., Domain and Task Modelling, Annual Progress Report 1990 SIMPR document No. SIMPR-NRC-1990-12I, December 1990.

96. Hätönen, K. Saarinen, P. Stickler, P. Tyrväinen, P., Domain and Task Modelling, System Specification, SIMPR document No. SIMPR-NRC-1990-9e, May 1990.

97. Hätönen, K. Saarinen, P. Stickler, P. Tyrväinen, P., Knowledge-Based Text Retrieval using Domain and Task Models, Finish Artificial Intelligence Days (STeP-90), Oulu, Finland. July 13, 1990.

98. Ingwersen, P., Information Retrieval Interaction, Taylor Graham, 1992.

99. Ingwersen, P. Wormell, I., More Indexing and Retrieval Techniques Matching Different Types of Information Needs, Proceedings of 44th FID Conference, Information Rederation for Information and Documentation, Vol. 1. 1988, pp. 192-203.

100. Jacobs, P.S. Rau, L.F., Natural Language Techniques for Intelligent Information Retrieval, ACM SIGIR, Proceedings of 11th International Conference on Research & Development in Information Retrieval, Grenoble, France, June 13-15, 1988, pp. 85-99.

101. Jacobs, P.S. Rau, L.F., SCISOR: Extracting Information from On-line News, Communications of the ACM, Vol. 33 (1990), No. 11, 1990, pp. 88-97,

102. Information technology - Text and office systems - Document Style Semantics and Specification Language (DSSSL), Draft International Standard, ISO/IEC DIS 10179. International Organization for Standardization, 1991.

103. Information technology - Text and office systems - Hypermedia/Time-based Document Structuring Language (HyTime), Draft International Standard, ISO/IEC DIS 10744. International Organization for Standardization, 1991.

104. Jones, K.S., A look back and a look forward, ACM SIGIR, Proceedings of 11th International Conference on Research & Development in Information Retrieval, Grenoble, France, June 13-15, 1988, pp. 13-29.

105. Jäppinen, H. Honkela, T. Lehtola, A. Valkonen, K., Hierarchical Multilevel Processing Model for Natural Language Database Interface, Proceedings of the Fourth IEEE Conference on Artificial Intelligence Applications, San Diego, CA. March 14-18 1988. pp. 332-337.

106. Kangassalo, H. Ohsuga, S. Jaakkola, H. (eds.), Information Modelling and Knowledge Bases, IOS Press, Amsterdam, 1990. pp. 264-291.

107. Karlsson, F. Voutilainen, A. Heikkilä, J. Anttila, A., Natural Language Processing for Information Retrieval Purposes, SIMPR Document No. SIMPR-RUCL-1990-13.4e, 1990.

108. Karlsson, F. Voutilainen, A. Anttila, A. Heikkilä, J., Constraint Grammar: a Language-Independent System for Parsing Unrestricted Text, with an Application to English, Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991.

109. Kerschenberg, L., Expert Database Systems: Knowledge/Data Management Environments for Intelligent Information Systems, Information Systems, Vol. 15 (1990), No. 1, pp. 151-160,

110. Kilpeläinen, P. Lindén, G. Mannila, H. Nikunen, E., A structured document database system, in Furuta, R. (ed.), Proceedings of EP90, the International Conference on Electronic Publishing, Document Manipulation & Typography, The Cambridge Series on Electronic Publishing, University Press, Cambridge, 1990, pp. 139--151.

111. Kim, Y.W. Kim, J.H., A Model of Knowledge Based Information Retrieval with Hierarchical Concept Graph, Journal of Documentation, Vol. 46 (1990), No. 2, pp. 113-136.

112. Kimoto, H. Toshiaki, I., Construction of a Dynamic Thesaurus and Its Use for Associated Information Retrieval, Proceedings of 3rd International Conference on Research & Development in Information Retrieval, 1990. pp. 227-240,

113. King, D.W. Griffiths, J.-M., Indicators of the use, usefulness and value of scientific and technical information, Proceedings of Online Information 91, London, England, December 10-12, 1991. Learned Information Ltd. pp. 361-377.

114. King, M. Perschke, S., EUROTRA and Its Objectives, Multilingua Vol. 1 (1982), No. 1, pp. 27-32.

115. Kleinbart, P., Prolegomenon to 'Intelligent' Thesaurus Software, Journal of Information Science (1985), NO. 11, pp. 45-53.

116. Klinker, G., A Framework for Knowledge Acquisition, Third European Workshop on Knowledge Acquisition for Knowledge-Based Systems (EKAW89), Paris, July 1989, pp. 102-116.

117. Kornwachs, K., System Knowledge, Contexts and Natural Language, in Elzas, M.S. Ören, T.I. Zeigler, B.P. (eds), Modelling and Simulation Methodology Knowledge Systems' Paradigms, Elsevier Science Publishers B.V. North-Holland, 1989.

118. Kotteman, J.E. Gordon, M.D. Stott, J.W., A Storage and Access Manager for ILL-STRUCTURED Data, Communications of the ACM, Vol. 34 (1991), No. 8, pp. 94-103.

119. Kotzias, K., How to respond to different language particularities by indexing texts using automatic text analysis, Proceedings of Online Information 90, London, England. 11-13 December 1990, pp. 61-68.

120. Koulopoulos, T.M., Cost Justifying Document Management, Text Management Journal, Delphi Consulting Group, Inc. Vol. 1 (1990), Issue 2, pp. 1,12-13.

121. Koulopoulos, T.M. (ed.), Handbook of Electronic Document Management Systems Evaluation & Design, Delphi Consulting Group. Boston, MA. 1991.

122. Kristensen, J. Järvelin, K., The Effectiveness of a Searching Thesaurus in Free-Text Searching in a Full-Text Database, Int. Classif. Vol. 17 (1990), No. 2, pp. 77-84.

123. Kuhlen, R. Yetim, F., HYPER-TOPIC - a system for the automatic construction of a hypertext-base with intertextual relations, Universität Konstanz, Dept. Information Science, P O Box 5560, W. Germany. July 1989.

124. Lancaster, F.W., Vocabulary Control FOR INFORMATION RETRIEVAL, (2nd ed.) Information Resources Press, Arlington, Virginia, 1986.

125. Lantz, B.E., The Relationship Between Documents and Relevant References, Journal of Documentation, 37. pp. 134-145.

126. Lebowitz, M., The Use of Memory in Text Processing, Communications of the ACM, Vol. 31 (1988), No. 12, pp. 1483-1502.

127. Ledwith, R.H., Development of a large, Concept-Oriented Database for Information Retrieval, ACM SIGIR, Proceedings of 11th International Conference on Research & Development in Information Retrieval, Grenoble, France, June 13-15, 1988. pp. 651-661.

128. Lehnet, W. Sundheim, B., A Performance Evaluation of Text-Analysis Technologies, AI Magazine, Vol. 12 (1991), No. 3, pp. 81-94.

129. Lenat, D.B. Guha, R.V. Pittman, K. Pratt, D. Shepherd, M., CYC: Toward Programs with Common Sense, Communications of the ACM, Vol. 33 (1990), No. 8, pp. 30-49.

130. Lewis, D.D. Croft, B. Bhandaru, N., Language-Oriented Information Retrieval, International Journal of Intelligent Systems, Vol. 4 (1989), No. 2, pp. 285-318.

131. Littlejohn, S.W., Theories of Human Communication, 3rd ed. Wadsworth Publishing Company, Belmont, California. 1989.

132. Locke, C., The Dark Side of DIP, Byte, April 1991. pp. 193-204.

133. Losee, R.M., An Analytic Measure Predicting Information Retrieval System Performance, Information Processing & Management, Vol. 27 (1991), No. 1, pp. 1-13.

134. Lung, A.C., Knowledge-Based Retrieval as the First Step of Knowledge Acquisition, Proceedings of 3th International Symposium on Knowledge Engineering, Madrid. SP, 1988, pp. 305-312,

135. Lytinen, S.T., Semantics-first Natural Language Processing, Proceedings of the Ninth National Conference on Artificial Intelligence, AAAI-91, AAAI Press / The MIT Press, Vol. 1, July 14-19, 1991. pp. 111-116.

136. Malone, L.C. Driscoll, J.R. Pepe, J.W., Modelling the Performance of an Automated Keywording System, Information Processing & Management, Vol. 27 (1991), No. 2/3, pp. 145-151.

137. Mann, J., Software To Manage The Paper Mountain, Datamation, July 15, 1991. pp. 79-80.

138. Mannila, H. Räihä, K-J., On query languages for the p-string data model, in Kangassalo, H. Ohsuga, S. Jaakkola, H. (ed.), Information Modelling and Knowledge Bases, IOS Press, Amsterdam, 1990. pp. 469-482.

139. Marchionini, G. Shneiderman, B., Finding Facts vs. Browsing Knowledge in Hypertext Systems, Computer, January 1988, pp. 70-80.

140. Marcus, A., Graphic Design for Electronic Documents and User Interfaces, ACM Press, Tutorial Series, Addison-Wesley, 1992.

141. Marinov, V., Referencing Manuals in English, Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991.

142. Mark, D.M., Cognitive Image-Schemata for Geographic Information: Relations to User Views and GIS Interfaces, Proceedings of GIS/LIS '89, Orlando, FL, USA, Vol. 2, 1989. pp. 551-560.

143. Maybury, M.T., Future Directions in Natural Language Processing; The Bolt Beranek and Newman Natural Language Symposium, AI Magazine, Summer 1990. pp. 12-13.

144. Mayer, G. Yamamoto, C. Evens, M. Michael, J.A., Constructing a Knowledge Base from a Natural Language Text, Proceedings of the 2nd Annual IEEE Symposium on Computer-Based Medical Systems, June 1989. pp. 98-107.

145. Mayfield, J. Nicholas, C.K., SNITCH: Augementing Hypertext Documents with a Semantic Net, in Yesha Y. (ed.), Information and Knowledge Management, CIKM-92, Publication of ISMM, ISBN: 1-880843-03-X, 1992. pp. 146-152.

146. McCawley, J.D., Everything that Linguists have always Wanted to Know about Logic* *but were ashamed to ask, University of Chicago, 1984 (1981).

147. McKell, P., Intelligent Documentation; From Word Processing To Knowledge Processing, Knowledge Based Systems Management Review, Vol 2 (1990), No. 3, pp. 3-6.

148. McMath, C.F. Tamaru, R.S. Rada, R., A graphical thesaurus-based information retrieval system, International Journal of Man-Machine Studies (1989), 31, pp. 121-147.

149. Meersman, R.A. Sernadas, A.C. (eds.) Data and Knowledge (DS-2), Second IFIP 2.6 Working Conference on Database Semantics, 'Data and Knowledge' (DS-2), North-Holland, 1988.

150. Michalski, G.P., The World of Documents, Byte, April 1991. pp. 159-170.

151. Minsky, M., A Framework for Representing Knowledge, in Winston, P.H. (ed.), The Psychology of Computer Vision, McGraw-Hill, New York, 1975, pp. 211-277.

152. Minsky, M., Logical Versus Analogical or Symbolic Versus Connectionist or Neat Versus Scruffy, AI magazine, Vol. 12 (1991), No. 2, pp. 35-51.

153. Miyamoto, S. Oi, K. Abe, O. Katsuya, A. Nakayama, K., Directed Graph Representations of Association Structures: A Systematic Approach, IEEE Transactions on Systems, Man and Cybernetics Vol. 16 (1986), No.1, pp. 53-61.

154. Myler, H.R. Gonzalez, A.J., Automated Knowledge Base Generation from CAD Databases using Relaxation Techniques, Proceedings of the IEEE International Symposium on Intelligent Control. Arlington, VA, USA. August 1988, (IEEE Comput. Soc. Press 1989), pp. 211-214.

155. Nakamura, Y., Expression of Concepts in a Classification Systems, Proceedings of the international congress of Terminology and Knowledge Engineering, INDEKS Verlag, Frankfurt/M, 1987, pp. 243-252.

156. Neches, R. Fikes, R. Finin, T. Gruber, T. Rasmesh, P. Senator, T. Swartout, W.R., Enabling Technology for Knowledge Sharing, AI Magazine, Vol. 12 (1991), No. 3, pp. 36-56.

157. Nelson, T.H., Replacing the Printed Word: A Complete Literary System, Proceedings of IFIP. October 1980, pp. 1013-1023.

158. Newcomb, S.R. Kipp, N.A. Newcomb, V.T., The 'HyTime' Hypermedia/Time-based Document Structuring Language, Communications of the ACM. Vol. 34 (1991), No. 11, pp. 67-83.

159. Nicolson, R.I. Tomlinson, P., USHIR: A Knowledge-Based Hypermedia System, Hypermedia, Vol. 3 (1991), No. 1, pp. 1-33.

160. Niemistö, J., Suomenkielisten tekstidokumenttien arkistointijärjestelmä, (title in English: Archiever for Finnish Text Documents), M.Sc. thesis, Helskini University of Technology, 1988.

161. Niemistö, J. Jäppinen, H., Some Linguistic Problems for Automatic Indexing caused by Inflextions and Compaunds, in Hämäläinen, P. Koskiala, S. Repo, A.J. (eds.), 44th FID Conference and Congress, Part I, International Federation for Information and Documentation, Finnish Society of Information Services, 1988. pp. 158-163.

162. Nishida, T. Kosaka, A. Doshita, S., Towards Knowledge Acquisition from Natural Language Texts: Automatic Model Construction from Hardware Manual, Proceedings of the 8th International Joint Conference on Artificial Intelligence, IJCAI-83, Los Altos, CA. W. Kaufmann. 1983. pp. 482-486.

163. O'Connor, L.J. Partridge, D.R. Dolan, C.P. Ebeid, N. Goddard, N.H., Automating Knowledge Acquisition - A Survey of Systems and Approaches, Proceedings of the second annual conference on Expert Systems, April 1988, pp. 181-201.

164. Pao, M.L., Retrieval Differences between Term and Citation Indexing, in Hämäläinen, P. Koskiala, S. Repo, A.J. (eds.), 44th FID Conference and Congress,. Part I. International Federation for Information and Documentation, Finnish Society of Information Services, 1988, pp. 217-224

165. Parsaye, K. Chignell, M. Khoshafian, S. Wong, H., Intelligent Databases: Object-Oriented, Deductive and Hypermedia Technologies, New York, Willey, 1989.

166. Partridge, S.K., So What Is Task Orientation, Anyway? IEEE Transactions on Professional Communications, Vol. 29 (1986), No. 4, pp. 26-32.

167. Patel-Schneider, P.F. Brachman, R.J. Levesque, H.J., ARGON: Knowledge Representation meets Information Retrieval, IEEE Computer, December 1984, pp. 280-286.

168. Pazienza, M.T. Velardi, P., Using a Semantic Knowledge Base to Support a Natural Language Interface to a Text Database, Entity-Relationship Approach: A Bridge to the User. Proceedings of the 7th International Conference. Rome, Italy, November 16-18, 1988, pp. 313-330.

169. Pedersen, G.S. (ed.), SIMPR, Structured Information Management: Processing and Retrieval, Annual Progress Report 1989, SIMPR document No. SIMPR-CRI-1990-1.1e, January 1990.

170. Peltason, C., The BACK System - An Overview, SIGART Bulletin, Special Issue on Implemented Knowledge Representation and Reasoning Systems, Vol. 2 (1991), No. 3, pp. 114-119.

171. Pinelli, T.E. Glassman, W.E. Oliu, W.E. Barclay, R.O., Technical Communications in Aeronautics: Results of an Exploratory Study, NASAS TM-101534, 1989.

172. Pollit, A.S., Intelligent Interfaces to Online Databases, Expert Systems Information Management, Vol. 3 (1990), No. 1, pp. 49-69.

173. Postma, G.J. Van der Linden, B. Smits, J.R.M. Kateman, G., TICA: A System for the Extraction of Data from Analytical Chemical Text, Chemometrics and Intelligent Laboratory Systems, (1990), 9, pp. 65-74.

174. Puscas, M.L., A Survey of Technical Computer Users Resulting in Guidelines for the Development of Technical Computer Documentation, Proceedings of ACM SIGDOC89, Pittsburgh, Pennsylvania. November 8-10, 1989, pp. 49-65.

175. Rada, R., Augmenting Thesauri for Information Systems, ACM Transactions on Office and Information Systems, Vol. 5 (1987), No. 4, pp. 378-392.

176. Rada, R., HYPERTEXT: from Text to Expertext, McGraw-Hill Book Company, London, 1991.

177. Rada, R. Hafedh, M., A Knowledge-Intensive Learning System for Document Retrieval, in Morik, K. (ed.), Knowledge representation and organization in machine learning, 1989, pp. 65-87.

178. Rada, R. Mili, H. Bicknell, E. Blettner, M., Development and Application of a Metrics on Semantic Nets, IEEE Transactions on Systems, Man and Cybernetics, Vol. 19 (1989), No. 1, pp. 17-30.

179. Rasmussen, J. The Role of Hierarchical Knowledge Representation in Decision Making and System Management, IEEE Transactions on Systems, Man and Cybernetics 15 (1985), pp. 234-243.

180. Rau, L.F. Jacobs, P.S. Zernik, U., Information Extraction and Text Summarization Using Linguistic Knowledge Acquisition, Information Processing & Management, Vol. 25 (1989), No. 4, pp. 419-428.

181. Rettig, M., Nobody Reads Documentation, Commendations of the ACM. Vol. 34 (1991), No. 7, pp. 19-24.

182. Rumbaugh, J. Blaha, M. Premerlani, W. Eddy, W. Lorensen, W., Object-Oriented Modelling and Design, Prentice-Hall International, New Jersey, 1991.

183. Saarinen, P. Tyrväinen, P., Domain and Task Modelling, SIMPR document No. SIMPR-NRC-1989-10.8e, February 1990.

184. Salminen, A., Four Levels of a Document Database Model, in Kangassalo, H. Ohsuga, S. Jaakkola, H. (ed.), Information Modelling and Knowledge Bases, IOS Press, Amsterdam, 1990, pp. 211-238.

185. Salton, G., Another Look at Automatic Text retrieval Systems, Communications of the ACM, Vol. 29 (1986), No. 7, pp. 648-656.

186. Salton, G., Automatic Text Processing, Addison-Wesley, 1989.

187. Salton, G., Developments in Automatic Text Retrieval, Science, Vol. 253 (1991), pp. 974-980.

188. Salton, G. Buckley, C., Automatic Text Structuring and Retrieval - Experiments in Automatic Encyclopedia Searching, ACM SIGIR, Proceedings of 14th International Conference on Research and Development in Information Retrieval, Chicago, Illinois USA. October 13-16, 1991, pp. 21-29.

189. Salton, G. Buckley, C., Global Text Matching for Information Retrieval, Science, Vol. 253 (1991), pp. 1012-1015.

190. Salton, G. McGill, M.J., Introduction to Modern Information Retrieval, McGraw-Hill, New York, 1983.

191. Savitch, W. et.al. (eds.), The Formal Complexity of Natural Language, D. Reider Publishing Company, 1987.

192. Schwarz, C., Content Based Text Handling, Informatin Processing and Management, Vol. 26 (1990), No. 2, pp. 219-226.

193. Schäube, P., An Information Structure dealing with Term Dependence and Polysemy, ACM SIGIR, Proceedings of 11th International Conference on Research & Development in Information Retrieval, Grenoble, France, June 13-15, 1988. pp. 519-533.

194. Shapiro, S.C. et. al. (eds.) Encyclopedia of Artificial Intelligence, Vols. 1 & 2, John Wiley & Sons, 1990.

195. Sheridan, P., Syntactic Processing for Text Analysis: A Survey, SIMPR Document No. SIMPR-DCU-1989-02, Dublin City University, Ireland, November 1989, pp. 23.

196. Sheridan, P., Smeaton, A.F., Structured Analysis: A Method for Handling Ambiguity, SIMPR Document No. SIMPR-DCU-1990-16.1e, Dublin City University, Ireland, March 1990.

197. SIGART Bulletin, Special Issue on Implemented Knowledge Representation and Reasoning Systems, Vol. 2 (1991), No. 3.

198. Smart, G., Automatic Information Management Based on Linguistic Analysis - Speaking the same Language? Expanded notes for the presentation at First STINFON Conference, Nijmegen, The Netherlands, 18 December 1991. (Available from CRI A/S, Birgerod, Denmark).

199. Smeaton, A.F., van Rijsbergen, C.J., Experiments on Incorporating Syntactic Processing of User Queries into a Document Retrieval Strategy, ACM SIGIR, Proceedings of 11th International Conference on Research & Development in Information Retrieval, Grenoble, France, June 13-15, 1988, pp. 31-51.

200. Smeaton, A.F., SIMPR: Using Natural Language Processing Techniques for Information Retrieval, Proceedings of British Computer Society Information Retrieval Research Colloquium, Huddersfield, UK, April 1990.

201. Smeaton, A.F. Voutilainen, A. Sheridan, O., The Application of Morpho-Syntactic Language Processing to Effective Text Retrieval, Proceedings of the Annual ESPRIT Conference, Brussels, November 12-15, 1990, Kluwer, London, pp. 619-636.

202. Smith, L.C., Artificial Intelligence and Information Retrieval, in Williams, M.E. (ed.), Annual Review of Information Science and Technology, Elsevier, Amsterdam, Netherlands, Vol. 22 (1987), pp. 41-77.

203. Smith, P.J. Krawczak, D. Shute, S.J. Chinell, M.H., Bibliographic information retrieval systems: increasing cognitive compatibility, Information Services & Use, Vol. 7 (1987), No. 4-5, pp. 95-102.

204. Soergel, D., Organizing Information, Principles of Data Base and Retrieval Systems, Academic Press, Inc. Orlando, Florida. 1985.

205. Soergel, D., Indexing Languages and Thesauri: Construction and Maintenance, Wiley, New York, 1974.

206. Sowa, J.F., Conceptual Structures: Information Processing in Mind and Machine, Reading, MA, Addison-Wesley, 1984.

207. Spayd, M.K., The Cost of Knowledge, Text Management Journal, Delphi Consulting Group, Inc. Vol. 1 (1990), Issue 2, pp. 18-20.

208. Spayd, J.K., J-SPACE: A Knowledge Representation Format for Use in Natural Language Text Searching, Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991.

209. Sperberg-McQueen, C.M. Burnard, L. (eds.), ACH - ACL - ALLC, Guidelines For the Encoding and Interchange of Machine-Readable Texts, TEI P1, (Text Encoding Initiative), Draft Version 1.0, (Originally from Association for Computers and the Humanities, Poughkeepsie, NY). Chicago and Oxford, July 16, 1990.

210. Stefik, M. Bobrow, D., Object-oriented Programming: Themes and Variations, Ai Magazine, Winter 1986. pp. 40-62.

211. Stickler, P., On the Nature of Technical Documentation, SIMPR document No. SIMPR-NRC-1990-13i, December 1990.

212. Sullivan, J.W. Tyler, S.W. (eds.), Intelligent User Interfaces, ACM Press, Frontier Series, Addison Wesley, 1991.

213. Tague, J. Schultz, R., Some Measures and Prpcedures for Evaluation of the User Interface in an Information Retrieval Systems, Proceedings of the 11th ACMSIGIR Conference on Research & Developpment in Information Retrieval. ACM Press, 1988, pp. 371-385.

214. Tan, T.C. Smith, P., RADA - A Document Retrieval System Incorporating AI and Expert System Approaches, Proceedings of the 11th BCS IRSG Research Colloquium on Information Retrieval, Huddersfield, UK. 5-6 July 1989, pp. 3-15.

215. Taucher, W. Hospodarsky, J. Krause, J. Schneider, C. Womser-Hacker, C., Effects of linguistic functions on information retrieval in a German-language full-text database: comparison between retrieval in abstract and full text, Online Review, Vol. 15 (1991), No. 2, pp. 77-86.

216. Telecom 91, Digit international, Bulletio of Nokia Telecommunication, October 7, 1991.

217. Tenopir, C., Full-Text Databases, Annual Review of Information Science and Technology, 19, Williams, M. (ed.), Knowledge Industries Publications Inc., White Plains, NY. 1984, pp. 215-246.

218. Terveen, L.G. Wroblewski, D.A., A Tool for Achieving Consensus in Knowledge Representation, Proceedings of the Ninth National Conference on Artificial Intelligence, AAAI-91, AAAI Press / The MIT Press, Vol. 1, July 14-19, 1991, pp. 74-79.

219. Text Retrieval Systems, A Market and Technology Assessment, Vol. 1-3. Delphi Consulting Group, 1991

220. Thompson, R.H. Croft, W.B., Support for browsing in an intelligent text retrieval system, Man-Machine Studies (1989), No. 30, pp. 639-668.

221. Tong, R.M. Balcom, L.B., Focused Text Interpretation: A Mechanism for Controlling the Cost / Accuracy Tradeoff, Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991.

222. Tong, R.M. Appelbaum, L.A. Askim, V.N., A Knowledge Representation for Conceptual Information Retrieval, International Journal on Intelligent Systems, Vol. 4 (1989), No. 2, pp. 259-284.

223. Tsai, S-T. Yang, C-C. Lien C-C., Automated retrieval of consistent documentation for rapid prototyping systems and software maintenance Information and Software Technology, Vol. 32 (1990), No. 8, pp. 521-530.

224. Turtle, H. Croft, W.B., Evaluation of an Inference Network-Based Retrieval Model, ACM Transactions on Information Systems, Vol. 9 (1991), No. 3, pp. 187-222.

225. Tyrväinen P., Reusable Object-Oriented Tools and Their Applications: AIGT - An Object-Oriented Interface Tool Library, DMG - An Application for Model Based Diagnostics, Proceedings of First International Conference in Technology of Object-Oriented Languages and Systems (TOOLS'89), Paris, November 13-15, 1989, pp. 411-421.

226. Tyrväinen, P., DMG - Model Based Hypertext Generation for Practical Production of Diagnostic Advisory Systems, Proceedings of EUROPAL'90, the First European Conference on the Practical Application of Lisp, Cambridge, UK, March 27-29, 1990, pp. 329-339.

227. Tyrväinen, P., DMG - Object-Oriented Iterative Knowledge Acquisition for the Generation of Diagnostic Hypertext Systems, LISP Pointers, ACM Press, Vol. 4 (1991), No. 1.

228. Tyrväinen, P.T., Feasibility of NLP, Index Term Extraction, and Domain Modelling to Processing and Retrieval of Technical Documentation, Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991. (Also SIMPR Document No. SIMPR-NRC-1991-26-15E)

229. Tyrväinen, P., On the Measures for Problem-Oriented Information Retrieval Process, SIMPR Document No. SIMPR-NRC-1991-26-16e, September 1991.

230. Tyrväinen, P., Hypertext and Text retrieval, in Hyvönen, E. Seppänen, J. Syrjänen, M. (eds.), STep-92 New Directions in Artificial Intelligence, Vol. 2 - Symposia, Publications of the Finnish Artificial Intelligence Society (FAIS), 1992, pp. 96-105.

231. Tyrväinen, P., Domain and Task Modelling -- Background, Methodology, and Application, SIMPR Document no: SIMPR-NRC-1992-31.3E, Nokia Research Center, P.O.Box 156, FIN-02101 Espoo, Finland. May 21, 1992.

232. Tyrväinen, P. Saarinen, P. Hätönen, K., Domain Modelling for Technical Documentation Retrieval, in Kangassalo, H. Jaakkola, H. Hori, K. Kitahashi, T. (eds.), Information Modelling and Knowledge Bases IV, IOS Press, The Netherlands, 1993, pp. 388-399.

233. Tyrväinen, P. Saarinen, P. Hätönen, K., DTM -- Domain Modelling for Technical Documentation Retrieval, in Yesha Y. (ed.), Information and Knowledge Management.CIKM-92, Publication of ISMM, ISBN: 1-880843-03-X, 1992, pp. 509-516.

234. Utne, I., Technical Terminology at the Norwegian Term Bank - a Basis for Knowledge Representation and Information Retrieval, Proceedings of the international congress of Terminology and Knowledge Engineering, INDEKS Verlag, Frankfurt/M, 1987, pp. 357-367.

235. van Rijsbergen, C.J., Information Retrieval, Second Edition, Butterworths, London 1979.

236. Voorhees, E.M., The Effectiveness and Efficiency of Agglomerative Hierarchic Clustering in Document Retrieval, Doctoral Dissertation, Cornell University, Ithaca, NY, January 1986.

237. Waterworth, J.A. Chinell, M.H., A Model for Information Exploration, Hypermedia, Vol. 3 (1991), No. 1, pp. 35-58.

238. Waterman, D.A., A Guide to Expert Systems, Addison-Wesley Publishing Company, Inc. USA, 1986.

239. Watters, C.R., Logic Framework for Information Retrieval, Journal of the American Society for Information Science, Vol. 40 (1989), No. 5, pp. 311-324.

240. Weawer, M.T. France, R.K. Chen, Q-F. Fox, A., Using a Frame-Based Language for Information Retrieval, International Journal on Intelligent Systems, Vol. 4 (1989), No. 2, pp. 223-258.

241. Werner, P.R., Steps Toward Building Shared Information Cultures in Organizations, Natural Language Text Retrieval, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, California, July 15, 1991.

242. Wilson, E., Integrated Information Retrieval for Law in a Hypertext Environment, ACM SIGIR, Proceedings of 11th International Conference on Research & Development in Information Retrieval, Grenoble, France, June 13-15, 1988, pp. 663-677.

243. Woods, D.D. Johannesen, L. Potter, S.S., Human Interaction with Intelligent Systems: An Overview and Bibliography, Bibliographies and Literature Reviews, SIGART Bulletin, Vol. 2 (1991), No. 5, pp. 39-50.

244. Wright, P., Presenting Technical Information: A Survey of Research Findings, Information Science (1987), 6, pp. 93-134.

245. Wu, H., On Query Formulation in Information Retrieval, Doctoral Dissertation, Cornell University, Ithaca, NY, 1981.

246. Yesha Y. (ed.), Information and Knowledge Management.CIKM-92, Publication of ISMM, ISBN: 1-880843-03-X, 1992.

247. Zabezhalio, M.I., Scientific Information Systems and Prospects for New Information Technology, Automatic Documentation and Mathematic Linguistics, Vol. 22 (1988), No. 2, pp. 1-16.

248. Zarri, G.P., A Knowledge Representation Language for Large Knowledge Bases and 'Intelligent' Information Retrieval Systems, Information Processing & Management, Vol. 26 (1990), No. 3, pp. 349-370.

249. Zeigler, B. Rada, R., Abstraction in Methodology: A Framework for Computer Support, Information Processing and Management, Vol. 20 (1984). pp. 63-79.

250. Zdonik, S.B. and Maier, D. (eds.) Readings in Object-Oriented Database Systems, Morgan Kaufman Publishers, Inc., San Mateo, CA, 1990.

APPENDICES

APPENDIX A: DTM - INTERFACE TO DESIGN INFORMATION INPUT

APPENDIX B: DTM - CREATION OF TEXT REFERENCES