

MSc. Thesis

Basora Enterprise Search

A New System for Enterprise Information Retrieval

October 28, 2008



Programme

MSc. Media and Knowledge Engineering

Scientific institute

Technical University Delft

Organization

Logica B.V. Netherlands

Faculty

Electrical Engineering, Mathematics & Computer Science

Author

M.G. Wallé BSc. 1174843

logica

**TU Delft**
Delft University of Technology

MSc. Thesis: Basora Enterprise Search

A New System for Enterprise Information Retrieval

Tuesday, October 28, 2008

Maurice Gregory Wallé 1174843

Abstract

There are several issues at play within Enterprise Search, a mayor one is that enterprises are finding it difficult to retrieve their data using current Enterprise Search (ES) solutions. This thesis work is a step towards improving this. By designing, implementing and evaluating a system for Enterprise Information Retrieval (IR), with an improved search result presentation technique, which assists users during search tasks. This was accomplished by exploring current research within the IR domain, then researching current ES systems and search result presentation techniques. This research resulted in finding two forms of document summaries, namely a textual document summary technique: Top Ranking Sentences (TRS) and a visual document summary technique: Thumbnails. These document summaries have been designed to support a user during information seeking activity. A cognitive model for IR was also designed to get a better understanding of the information seeking process. The system developed was named Basora Enterprise Search (BES) and was developed using an agile software development approach. It incorporates the TRS and Thumbnail into its search result presentation technique. The BES prototype was put through both a performance and user evaluation. The first test indicated that BES performs more or less equal to the Commercial Enterprise Search Solution IBM Omnifind 8.5, while providing the user with two extra forms of document summaries. The user evaluation focussed on evaluation of the effectiveness of the new search result presentation technique. The results of the user evaluation show that there are various search tasks where the addition of these summary elements has a positive effect on relevance assessment and query reformulation. This research indicates that the BES system actually helps a user assess the relevance of a document, minimizing the amount of documents that need to be opened before the user finds the desired one. It also indicates that the additional visual and textual document summaries assist the user when reformulating a search query, decreasing the time it takes a user to complete a search task.

Keywords: Enterprise Search, Interfaces, Top Ranking Sentences, TRS, Thumbnails

Graduation Committee

Prof. Drs. dr. L.J.M. Rothkrantz, Ir. H.J.A.M. Geers,
Dr. ir. P. Wiggers, Drs. B. Vranken & Ir. E.C. Essenius

Master Programme

Media and Knowledge Engineering

Man-Machine Interaction Group

Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology, The Netherlands
<http://mmi.tudelft.nl/> last visited: 20 October 2008

Maurice Gregory Wallé
mauricewalle@gmail.com

Basora Enterprise Search

A New System for Enterprise Information Retrieval

MSc. Thesis
Tuesday, 28 October 2008

Man-Machine Interaction Group
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology, The Netherlands
<http://mmi.tudelft.nl/> last visited: 20 October 2008

Logica B.V.

Logica is a UK-based global IT and management company, that is quoted on the London Stock Exchange and on Euronext Amsterdam. As of 2008 the company is active in 41 countries and has nearly 39.000 employees. The company name had changed from Logica to LogicaCMG after a merger with the company CMG. To create a global brand after several mergers and buyouts the company has reverted to its original name “Logica” in February 2008. Logica’s mission statement entails:

“To become the most trusted innovation partner, creating sustainable business value by using our local insight and global talent”.

Working Tomorrow

Working Tomorrow is a program started by Logica which employs students at different locations to work on their final thesis. All of the projects are innovative in the field of technology, concept or method.

The Working Tomorrow program has 3 main objectives:

- Recruit employees
- Increase the reputation of Logica concerning innovation
- Use demos and prototypes in the negotiation trajectory of bigger projects.

This research project is placed in the division IDT-ECM. IDT stands for Industry Distribution and Transport, while ECM is an acronym for Enterprise Content Management. This division is responsible for all the work done within the Enterprise Search field. Figure 1 shows the organizational chart of Logica and Working Tomorrow.

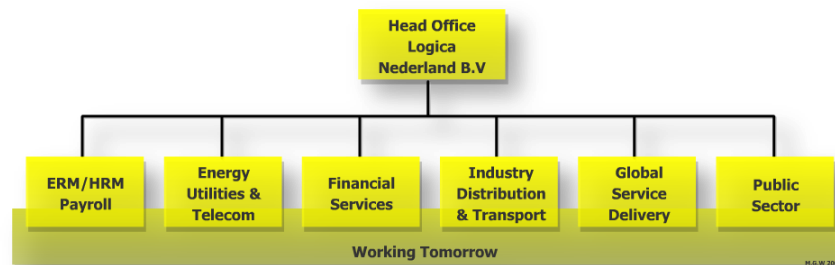


Figure 1: Logica Organizational Structure

Preface

It is tempting, if the only tool you have is a hammer, to treat everything as if it were a nail." - Abraham Maslow

Information Retrieval is an interesting research area in which to make a contribution, because it is a thriving field. Considering that my interests lie in this area, it was quite an evident choice as a research topic for my MSc. in Media and Knowledge Engineering.

The name of the Enterprise Search Engine designed, developed and evaluated during the course of this research, as stated in the title is Basora. This is a word in the language Papiamentu which means broom. The main idea behind this name is that the search engine sweeps away all the irrelevant results while leaving the relevant ones. Seeing that I am originally from Curacao, and this being our language it seemed appropriate. On the cover of my thesis a picture of a four-eyed butterfly fish is presented, this was taken by myself at Porto Marie bay on Curacao, this seemed suitable because of the four eyes, two of which are always seeking and or searching.

During the process of completing my research assignment and thesis work, I was always searching for information and or papers. What annoyed me were the search systems available at libraries and the web based paper archives. They were often inaccurate and some very difficult to work with to find exactly what I was looking for. This was a motivational factor which inspired me even more to finalize this research. Afterwards it is very rewarding to see how the system is actually being used by the interns at Logica and that they are very satisfied with the performance.

While working on my thesis work, I realized how useful the knowledge gathered from completing the Bachelor and Master subjects was. Some examples of these subjects being Data-structures and Algorithms, Internet Technology, Advanced Software Engineering, Intelligent User Experience Engineering, etc for designing and developing the prototype; while subjects like Statistical Data Analysis, Probability and Statistics came in handy for processing of the evaluation results.

It took about 400 commits, 20.000 lines of code and lots of obstacles were overcome to complete this research project. I will always make good use of all the knowledge, tools and experiences gathered. The quote at the top of this page always makes me appreciate all the opportunities I've had to educate myself further, and in doing so striving for not only a solution but the most optimal one.

Acknowledgements

I would like to thank my professor Leon Rothkrantz for his advice and guidance during the extent of the thesis work. Additionally I would like to thank Logica for their hospitality and the opportunity to carry out my research project and thesis work there. Instead of just implementing an existing Enterprise Search Solution Logica gave me the opportunity to broaden my knowledge and implement a custom solution whilst contributing to the ever evolving science of Information Retrieval. Furthermore I would like to thank Bram Vranken, Edwin Essenius, Ron Maarschalkerweerd, Michiel Cuijpers and Martin van Amersfoort for their help and guidance as well.

Last but certainly not least I would like to thank my parents and family for all the love and support, "Thank you all I appreciate it".

Maurice Wallé
Delft, the Netherlands

28th of October 2008

Contents

Logica B.V.....	5
Preface.....	7
1 Introduction.....	13
1.1 Problem Definition & Research Challenges.....	13
1.2 Research Topic.....	14
1.3 Motivation.....	15
1.4 Approach.....	15
1.5 Structure of the Document.....	16
2 Related Work.....	17
2.1 Current Research in IR.....	17
2.1.1 Formal Models.....	18
2.1.2 Design and Evaluation.....	18
2.1.3 Topic Identification and News Retrieval.....	19
2.1.4 User Interests and Workspaces.....	20
2.1.5 Clustering and Classification.....	21
2.1.6 Refinement and Feedback.....	21
2.1.7 Web Search.....	22
2.1.8 Structure/XML.....	23
2.1.9 Multimedia.....	23
2.2 Discussion.....	24
3 Selection from Related Work.....	27
3.1 Search Result Elements.....	27
3.1.1 Top Ranking Sentences.....	27
3.1.2 Thumbnail.....	28
3.1.3 TRS & Thumbnail Application.....	28
3.2 Anatomy of a Search Engine.....	29
3.3 Solr 1.2 Search Server.....	30
3.4 IBM Omnifind Search Solution.....	31
3.5 Discussion.....	32
4 Cognitive Model of IR.....	35
4.1 Search Model.....	35
4.1.1 A. User Defines a Query.....	36
4.1.2 B. Search.....	36
4.1.3 C. Display of Search Results.....	36
4.1.4 D. Interpretation and Refinement.....	37
4.1.5 Search Cycle Examples.....	38
5 Global Design.....	41
5.1 Methodology.....	41
5.2 Basora Enterprise Search Global Design.....	41
5.2.1 Basora Pre-Processing.....	42
5.2.2 Solr Indexer & Searcher.....	43
5.2.3 Basora Engine.....	43
5.2.4 Web Interface.....	43
5.2.5 TRS Algorithm.....	45
6 Implementation.....	47
6.1 Hardware, Software, Tools and Libraries.....	47

6.1.1	Server Side	47
6.1.2	Client Side.....	48
6.2	Basora Enterprise Search	48
6.2.1	Basora Pre-Processing	48
6.2.2	Solr Configuration	56
6.2.3	Basora Engine.....	57
6.2.4	Web Interface	58
6.2.5	Search Result Presentation Layouts.....	62
6.2.6	Basora TRS Calculation	63
6.2.7	Basora Searching.....	65
6.2.8	Future Additions.....	65
7	Evaluation	67
7.1	OS, Hardware & Software	67
7.2	Dataset	67
7.3	Performance Evaluation	69
7.4	User Evaluation	69
7.4.1	IMPACT.....	69
7.4.2	Intention	69
7.4.3	Metrics	70
7.4.4	People.....	72
7.4.5	Activities.....	72
7.4.6	Context.....	74
7.4.7	Technologies.....	74
7.4.8	Evaluation Plan.....	74
8	Evaluation Results	77
8.1	Performance Evaluation	77
	Discussion.....	79
8.2	User Evaluation.....	80
8.2.1	User Interaction.....	80
8.2.2	Relevance Assessment.....	82
8.2.3	Contribution of Layout Elements	83
8.2.4	Layout Preference.....	85
8.2.5	Participant Comments.....	86
9	Conclusions and Recommendations	87
9.1	Implications.....	88
9.2	Future Research	89
	Bibliography	91
	Books	91
	Research Assignment.....	91
	Papers	91
	Appendix.....	95
	Appendix A: Update Meeting Sheet	95
	Appendix B: Final Assessment Prototype.....	99
	Appendix C: Entry Questionnaire	103
	Appendix D: Session Questionnaire.....	103
	Appendix E: Exit Questionnaire	104
	Appendix F: Evaluation Assignment Form.....	105
	Appendix G: User Evaluation Procedure.....	110

List of Figures

Figure 1: Logica Organizational Structure	5
Figure 2: Google snippet example.....	27
Figure 3: File Browser Thumbnail Example.....	28
Figure 4: Search result presentation layout of system described in Joho and Jose 2006 [P2]	29
Figure 5: Basic search engine components.....	29
Figure 6: Components of the Google Search Engine (from Brin and Page 1998[P5]).....	30
Figure 7: Solr components.....	31
Figure 8: IBM Omnifind 8.5 Enterprise Search Interface	32
Figure 9: IBM Omnifind Result presentation	32
Figure 10: Cognitive Model: Search Cycle	35
Figure 11: Textual Document Summary - Top Ranking Sentences	37
Figure 12: Visual Document Summary - Thumbnail.....	37
Figure 13: Basora Enterprise Search prototype components.....	42
Figure 14: Basora main interface initial drawing.....	44
Figure 15: Concept Layout 1	44
Figure 16: Concept Layout 2	44
Figure 17: Concept Layout 3	44
Figure 18: Concept Layout 4	45
Figure 19: Crawler objects	49
Figure 20: File system.....	49
Figure 21: Class Diagram Crawler (The arrows indicate dependency).....	51
Figure 22: Parser Class Diagram (The arrows indicate dependency).....	53
Figure 23: Objects used by the parser (Getters and setters were left out)	54
Figure 24: Committer Class Diagram (The arrows indicate dependency).....	55
Figure 25: Basora Engine Controller Class Diagram	57
Figure 26: "ResultCell" Class.....	58
Figure 27: Basora Enterprise Search prototype main interface	59
Figure 28: Statistics interface	59
Figure 29: Help interface.....	60
Figure 30: About interface	60
Figure 31: Sorting and Filtering functions interface.....	61
Figure 32: Administrator Interface.....	61
Figure 33: Preferences Interface	62
Figure 34: Layout 1: Baseline	62
Figure 35: Layout 2: Baseline & TRS	62
Figure 36: Layout 3: Baseline & Thumbnail.....	62
Figure 37: Layout 4: Baseline, TRS & Thumbnail.....	63
Figure 38: TRS and Snippet Classes & Ranked Sentence Object.....	63
Figure 39: Pie Chart of the evaluation dataset	68
Figure 40: Basora Enterprise Search response time chart break-up.....	78
Figure 41: Basora vs. Omnifind response time.....	78
Figure 42: Basora vs. Omnifind results obtained per search query.....	78
Figure 43: Participants.....	80
Figure 44: User Interaction	81
Figure 45: Relevance Assessment	83

Figure 46: Contribution of layout elements 84
Figure 47: Layout preference 86

1 Introduction

Information Retrieval began with the birth of computers and the need to find data stored on these machines. Modern search technology has evolved from the simple search tools available on early operating systems. These tools emerged from the need to find data located on these systems quickly. Information Retrieval can be carried out in several environments, two of these are: Web and Enterprise. For each of these environments a different type of Search Engine is required namely Web Search Engine or Enterprise Search Engine. Web Search entails, using a search engine to identify and retrieve information housed on the World Wide Web, while Enterprise Search focuses on retrieval of information within Enterprises (i.e. Company's). This project focuses on the Enterprise Search environment.

The following paragraphs will give an introduction to this thesis work. First the problem definition is elucidated, and then the main research question is presented. Hereafter the motivation for choosing this topic is clarified. Next the approach taken is described, and finally the structure of the document is presented.

1.1 Problem Definition & Research Challenges

A term within Enterprise Search that is getting a lot of attention is "Findability". Its definition according to Frappaolo & Keldsen 2008 [P1] is "The Art and Science of making Content Findable". The reason for this attention is that the volume of content within businesses is growing at a phenomenal pace, and finding relevant content using simple queries is not sufficient, so the need arose for Enterprise search to become a science hence the term "Findability". Frappaolo & Keldsen 2008 [P1] state that the ineffectiveness of Enterprise Findability is not the fault of a poor Search Engine but the design behind its deployment is flawed.

When looking at the Enterprise Search domain from a different angle it is possible to identify several ontology's namely the user, the system and the network. Each of these has its own set of hurdles, within the first ontology the user; there are cognitive aspects like, how a user defines a query, how does a user assess the relevance of a document etc; here the user interface plays an important role.

The system has its own hurdles, such as how to convert a user's query into application language so the system understands what to retrieve based on the users query. Another issue is how to represent and store the data for the search engine to access this quickly, keeping the response time to a minimal.

The last ontology to consider is the network, one of the reasons is that network lag can seriously affect the response time of the Search Engine. It is also important to consider on what kind of system the search engine runs, because when running it from a PDA the network response time is slower than when using a computer. This research project focuses on the user and system aspect of Enterprise Search. There are three Enterprise Search issues that lie at the centre of this work:


- The first one is that *formulating and re-formulating good search queries* is proved to be a cognitively challenging task for users. These search queries are often approximations of a user's underlying need, making it difficult to formulate a query in one try, to satisfy this need; consequently making the information seeking process an iterative one [P2, P7, P8].
- The second one is *interpreting and assessing the relevance of documents* in search result list, this is also imperative to the search process [P2, P9]. Users are reluctant to examine large numbers of individual documents and they hardly look further than the first result page [P2, P10]. So a user's decision to view a document or not is totally dependent on the available elements within the search results namely: document name, author, file type, snippets, etc.

- Lastly *building a system* that allows a user to search through a dataset, which assists with reformulating search queries and interpreting and assessing the relevance of documents retrieved.

The first two issues are user related, while the last is associated with the system ontology. The following paragraph presents the research topic.

1.2 Research Topic

Before a research topic was formulated an extensive research assignment [RA] on current research areas within Information Retrieval was performed. This was done in order to give a good overview of current research and to use this knowledge to find an interesting research topic namely

Design, implement and evaluate a system for Enterprise Information Retrieval, which employs an improved search result presentation technique.  **BASORA**
ENTERPRISE SEARCH

The system created is called Basora Enterprise Search. Basora is a word in the language Papiamentu which means broom. The main idea behind this name is that the search engine sweeps away all the irrelevant results while leaving the relevant ones. During the initial research assignment [RA], the future research chapter indicated that Joho & Jose [P2] developed and evaluated a system which added various elements to the search result user interface for Web Search. This system's ingenuity was the inspiration for the development of the Basora Enterprise Search (BES) prototype it is discussed in chapter 3. In order to tackle this research topic the following research challenges were encountered:

- Search and Retrieval
- Presentation of Search Results
- Defining and Refining a User Query
- Cognitive Model of Information Retrieval
- User Validation of the System.

Each of these research challenges are discussed within this thesis. The first three challenges are discussed in chapter 2 and 3, while the third one is presented in chapter 4. The last challenge is explored in the chapters 7 and 8.

Now the two main Enterprise Search issues which the new system for Enterprise Information Retrieval addresses (as mentioned in paragraph 1.1), can be summarized as, facilitating a user during query reformulation, and improving their relevance assessment. This is tackled by focusing on the user interface component of the Search Engine. The main improvement is the search result presentation technique. Here two additions are made a visual and a textual document summary namely Top Ranking Sentences (TRS) and a Thumbnail. TRS are up to three sentences presented in the search results that contain the query terms and are ranked according to several features. Their main purpose is offering the user extra information about a document so beforehand they can form a better opinion of a document's contents [P3]. The thumbnail is a miniature image of the first page of a document, this increases ability of a user to make more accurate decisions about the relevance of results [P4]. These elements are further discussed in paragraph 6.2.5. The following paragraph gives some insight on the motivation behind this research.

1.3 Motivation

When looking for information, a search system can be a useful tool. Its interface is a means through which users interact with the search engine and control all aspects of their search. A search engines main purpose is to help its users find information that is helpful or relevant in order to complete a task. From a cognitive perspective it is logical that when a user is searching for information to complete a task the user lacks some knowledge, and information is needed to fill this knowledge gap [P2, P7]. Finding this information may entail executing several search queries, judging the relevance of the documents returned and opening & reading many documents. Considering all this it makes it very important to design and develop user interfaces which optimize the amount of information users obtain during a search; while presenting the information in a user friendly manner. This will facilitate the process of finding the necessary information as quickly and efficiently as possible, which is one of the main goals of this research.

Another interesting point is, that the terms used when submitting a search query are commonly approximations of the searchers underlying information need [P2, P8]. So the system returns results based on these search queries, and the usefulness and or relevance of these documents is not certain. Often documents contain partial information or redundant information, therefore in order to carry out an effective search, it is essential for users to be able to make sensible judgements about the documents contents, from the information provided to them in the search results. The only way to accomplish this is again to optimize the amount of information provided in the search result user interface and also providing this in a user friendly manner.

Working Tomorrow is a programme within Logica where interns get a chance, to carry out a project as a finalization of their Bachelor or Master Degree (as mentioned on page 4). Within the Working Tomorrow programme, the interns have access to an Enterprise Content Management system where they can store their documents, but the Enterprise Search system available performs very poorly. The reason this has to function properly is because about 400 projects have been executed by interns within Logica's Working Tomorrow programme and they have generated over 6000 documents related to their work. As new interns start their projects, accessing the work of others is vital to ensure that their project is unique, and this can also help them gather information on a certain topic. The interns may be able to use documents produced by their predecessors as valuable introductory reading which saves time on researching a topic themselves. It also helps the managers and architects, which have the responsibility of generating new and innovative projects by giving them easy access to previous and current interns thesis work, research assignment's, fact sheets, etc. This shows how important an Enterprise Search system can be within an organisation. This was another motivational factor for this research. The following paragraph outlines the approach taken during this research project.

1.4 Approach

This paragraph gives an overview of the approach used during the process of this research project. In order to design, develop and evaluate the proposed Enterprise Information Retrieval system the following steps were taken:

- Research related work on result presentation elements TRS and Thumbnail
 - What are the advantages?
 - How can they be implemented?
- Research on developing an Enterprise Search Engine
 - What components are needed?

- What components are ready to use and what still needs to be developed?
- Design and Development of an Enterprise Search prototype
 - Agile approach
 - Design new result presentation layouts
 - Weekly reviews of what has been developed and what was still needed
- Evaluation of the prototype
 - Performance test: Check the performance of the prototype
 - User test using IMPACT model to evaluate effectiveness of the new layouts
- Process User Test results
 - Calculate Averages, Standard Deviations, etc
 - Create Tables and Figures.

The first two steps of the approach consisted of exploring work related to this research and is referred to as the “Research Phase”. For the first step research was done on what the advantages are of the textual and visual result presentation elements and how to implement these. Then for the second step extensive research was performed in order to gain the knowledge needed to build an Enterprise Search Engine. Questions like “What components are needed to built it?”, “What components have already been developed and which are ready to use?”, “What programming language is best to use?”, and “What still needs to be developed and how?” were answered.

After completing this Research Phase, the Design and Development Phase started where the prototype was developed using an agile approach. Every week a few requirements were set, and worked on. At the end of the week the requirements were reviewed and the uncompleted ones were moved to the next week. Additional requirements were added when needed. Every 3 weeks an update meeting was planned with the stakeholders at Logica. Where the progress is discussed and they had a chance to add requirements and make comments. Logica was both the client and the end-user of the prototype.

During the Evaluation Phase the prototype was assessed to verify whether these visual and textual elements contributed to relevance assessment and query reformulation (the main Enterprise Search issues addressed with this research, presented in paragraph 1.3). An evaluation procedure was set up using IMPACT, this is a model for user evaluation described in paragraph 7.4.1. Then this procedure was carried out using 32 participants namely: 4 experts, and 28 regular users. Hereafter the results were processed and from these could be determined whether the search result presentation technique, indicates actual improvements.

The system developed is a proof of concept and therefore does not have all features of an enterprise search engine but only the ones necessary to ensure a full user experience. The next paragraph gives an overview of the document.

1.5 Structure of the Document

As mentioned in the previous paragraph this project went through several phases namely a Research Phase, a Design and Development Phase, and finally an Evaluation Phase. Each of these is discussed in the following chapters. Chapter 2, 3 and 4 describe the Research Phase of this project; it gives an overview of research within IR and also specific research related to this work. Next chapter 5 & 6 discuss the Design and Development phase; here the Global Design and Implementation of the developed prototype is elucidated. Then chapter 7 & 8 describe the Evaluation Phase of this research; these chapters explain the performance and user evaluation that was performed along with the results obtained. Finally the last chapter presents the conclusions and recommendations.

2 Related Work

Before starting the actual research on designing, developing and evaluating the new system for Enterprise Information Retrieval, some general research was carried out within the IR domain [RA]; this was mentioned in paragraph 1.2. The papers that were reviewed within this research assignment were each categorised. The following categories were used: Formal Models, Design and Evaluation, Topic Identification and News Retrieval, User Interests and Workspaces, Clustering and Classification, Refinement and Feedback, Web Search, Structure/XML and Multimedia. These categories are based on the ones used in Lalmas *et al* 2006 [P27], this paper gives an overview of work done within the IR domain in 2006. Not all the categories used in this paper were re-used because not all these categories are relevant seeing that more recent work from 2007 is also reviewed. The goal of this assignment is to give a broad overview of all recent development within the Information Retrieval Research domain. This chapter focuses on the essence of this research assignment, which formed the foundation of this thesis work, not all the papers are discussed in this chapter.

2.1 Current Research in IR

This paragraph covers recent research concerning Information Retrieval. Since quite some of research has been done in this field it is imperative to keep to a specific scope when looking for literature concerning search, in this case research from 2006 to present. By utilizing the library of the TUDelft, a variety of papers and books regarding these topics were attained. These were discussed through-out the research assignment [RA]. The papers that were reviewed are all from the 28th or 29th European Conference on Information Retrieval (IR) Research. The papers dated 2006 are from the 28th European Conference on IR Research, ECIR 2006, held in London, UK, on April 10-12, 2006. While the papers dated 2007 are from the: 29th European Conference on IR Research, ECIR 2007, held in Rome, Italy, on April 2-5, 2007. The papers were presented each in their corresponding category; additionally for every individual paper the abstract was given followed by a critical comments section and finalized with a table indicating what features the paper satisfies. The features that were used to evaluate the paper are the shown in Table 1 below.

Table 1: Features

<i>Feature</i>	<i>Clarification</i>
Broad Topic Area	Does the research scope of the paper cover a broad domain?
Model Based	Is the research based on an existing model?
Implemented	Has the research lead to an implementation?
Operational	Is the implementation working?
Evaluated	Has the implementation been evaluated?

Table 2 gives an overview of the papers that were reviewed in the research assignment [RA]. A “+” indicates that this feature is satisfied within the target paper, and a “-” indicates that it is not satisfied. An example being: for the paper Wang *et al* 2006. This paper does not cover a broad topic area, it is model based, and it’s been implemented, is operational and has been evaluated. The following paragraphs give an overview of the papers that were reviewed.

2.1.1 Formal Models

Wang *et al* 2006

Abstract [P13]

Implicit acquisition of user preferences makes log-based collaborative filtering favourable in practice to accomplish recommendations. In this paper, we follow a formal approach in text retrieval to reformulate the problem. Based on the classic probability ranking principle, we propose a probabilistic user-item relevance model. Under this formal model, we show that user-based and item-based approaches are only two different factorizations with different independence assumptions. Moreover, we show that smoothing is an important aspect to estimate the parameters of the models due to data scarcity. By adding linear interpolation smoothing, the proposed model gives a probabilistic justification of using TF×IDF-like item ranking in collaborative filtering. Besides giving the insight understanding of the problem of collaborative filtering, we also show experiments in which the proposed method provides a better recommendation performance on a music play-list data set.

McDonald 2007

Abstract [P14]

In this work we study the theoretical and empirical properties of various global inference algorithms for multi-document summarization. We start by defining a general framework for inference in summarization. We then present three algorithms: The first is a greedy approximate method, the second a dynamic programming approach based on solutions to the knapsack problem, and the third is an exact algorithm that uses an Integer Linear Programming formulation of the problem. We empirically evaluate all three algorithms and show that, relative to the exact solution, the dynamic programming algorithm provides near optimal results with preferable scaling properties.

2.1.2 Design and Evaluation

Demartini and Mizzaro 2006

Abstract [P15]

Effectiveness is a primary concern in the information retrieval (IR) field. Various metrics for IR effectiveness have been proposed in the past; we take into account all the 44 metrics we are aware of, classifying them into a two-dimensional grid. The classification is based on the notions of relevance, i.e., if (or how much) a document is relevant, and retrieval, i.e., if (how much) a document is retrieved. To our knowledge, no similar classification has been proposed so far.

Mooney *et al* 2006

Abstract [P16]

Current information retrieval systems make no measurement of the user's response to the searching process or the information itself. Existing psychological studies show that subjects exhibit measurable physiological responses when carrying out certain tasks, e.g. when viewing images, which generally result in heightened emotional states. We find that users exhibit measurable biometric behaviour in the form of galvanic skin response when watching movies, and engaging in interactive tasks. We examine how this data might be exploited in the indexing of data for search and within the search process itself.

Chernov *et al* 2007

Abstract [P17]

In the last years several top-quality papers utilized temporary Desktop data and/or browsing activity logs for experimental evaluation. Building a common test-bed for the Personal Information Management community is thus becoming an indispensable task. In this paper we present a possible dataset design and discuss the means to create it.

Yeung *et al* 2007

Abstract [P18]

Retrieval accuracy can be improved by considering which document type should be filtered out and which should be ranked higher in the result list. Hence, document type can be used as a key factor for building a re-ranking retrieval model. The authors take a simple approach for considering document type in the retrieval process. They adapt the BM25 scoring function to weight term frequency based on the document type and take the Bayesian approach to estimate the appropriate weight for each type. Experimental results show that the authors approach improves on search precision by as much as 19%.

2.1.3 Topic Identification and News Retrieval

Yao *et al* 2006

Abstract [P19]

Reading news is one of the most popular activities when people surf the internet. As too many news sources provide independent news information and each has its own preference, detecting unbiased important news might be very useful for users to keep up to date with what are happening in the world. In this paper we present a novel method to identify important news in web environment which consists of diversified online news sites. We observe that a piece of important news generally occupies visually significant place in some homepage of a news site and import news event will be reported by many news sites. To explore these two properties, we model the relationship between homepages, news and latent events by a tripartite graph, and present an algorithm to identify important news in this model. Based on this algorithm, we implement a system TOPSTORY to dynamically generate homepages for users to browse important news reports. Our experimental study indicates the effectiveness of proposed approach.

Shah and Eguchi 2007

Abstract [P20]

Several information organization, access, and filtering systems can benefit from different kind of document representations than those used in traditional Information Retrieval (IR). Topic Detection and Tracking (TDT) is an example of such a domain. In this paper we demonstrate that traditional methods for term weighing does not capture topical information and this leads to inadequate representation of documents for TDT applications. We present various hypotheses regarding the factors that can help in improving the document representation for Story Link Detection (SLD) - a core task of TDT. These hypotheses are tested using various TDT corpora. From our experiments and analysis we found that in order to obtain a faithful representation of documents in TDT domain, we not only need to capture a term's importance in traditional IR sense, but also evaluate its topical behaviour. Along with defining this behaviour, we propose a novel measure that captures a term's importance at the corpus level as well as its discriminating power for topics. This new measure leads to a much better document representation as reflected by the significant improvements in the results.

2.1.4 User Interests and Workspaces

Bogers and van den Bosch 2006

Abstract [P21]

We examine the use of authorship information in information retrieval for closed communities by extracting expert rankings for queries. We demonstrate that these rankings can be used to re-rank baseline search results and improve performance significantly. We also perform experiments in which we base expertise ratings only on first authors or on all except the final authors, and find that these limitations do not further improve our re-ranking method.

Vildjiounaite and Kallio 2007

Abstract [P22]

This works presents a method for explicit acquisition of context dependent user preferences (preferences which change depending on a user situation, e.g., higher interest in outdoor activities if it is sunny than if it is raining) for Smart Home – intelligent environment, which recognises contexts of its inhabitants (such as presence of people, activities, events, weather etc) via home and mobile devices and provides personalized proactive support to the users. Since a set of personally important situations, which affect user preferences, is user-dependent, and since many situations can be described only in fuzzy terms, we provide users with an easy way to develop personal context ontology and to map it fuzzily into common ontology via GUI. Backward mapping, by estimating the probability of occurrence of a user-defined situation, allows retrieval of preferences from all components of the user model.

2.1.5 Clustering and Classification

Osinski 2006

Abstract [P23]

In this paper we show how approximate matrix factorisations can be used to organise document summaries returned by a search engine into meaningful thematic categories. We compare four different factorisations (SVD, NMF, LNMF and K-Means/Concept Decomposition) with respect to topic separation capability, outlier detection and label quality. We also compare our approach with two other clustering algorithms: Suffix Tree Clustering (STC) and Tolerance Rough Set Clustering (TRC). For our experiments we use the standard merge-then cluster approach based on the Open Directory Project web catalogue as a source of human-clustered document summaries.

Sevillano *et al* 2007

Abstract [P24]

A major problem encountered by text clustering practitioners is the difficulty of determining a priori which is the optimal text representation and clustering technique for a given clustering problem. As a step towards building robust document partitioning systems, we present a strategy based on a hierarchical consensus clustering architecture that operates on a wide diversity of document representations and partitions. The conducted experiments show that the proposed method is capable of yielding a consensus clustering that is comparable to the best individual clustering available even in the presence of a large number of poor individual labelings, outperforming classic non-hierarchical consensus approaches in terms of performance and computational cost.

2.1.6 Refinement and Feedback

Yamout *et al* 2006

Abstract [P25]

A new Relevance Feedback (RF) technique is developed to improve upon the efficiency and performance of existing techniques. This is based on propagating positive and negative weights from documents judged relevant and not relevant respectively, to other documents, which are deemed similar according to one of a number of criteria. The performance and efficiency improve since the documents are treated as independent vectors rather than being merged into a single vector as is the case with traditional approaches, and only the documents considered in a given neighbourhood are inspected. This is especially important when using large test collections.

Roulland *et al* 2007

Abstract [P26]

We have developed an interactive query refinement tool that helps users search a knowledge base for solutions to problems with electronic equipment. The system is targeted towards non-technical users, who are often unable to formulate precise problem descriptions on their own. Two distinct but interrelated functionalities support the refinement of a vague, non-technical initial query into a more precise problem description: a synonymy mechanism that allows the system to match non-technical words in the query with corresponding technical terms in the knowledge base, and a novel refinement mechanism that helps the user build up successively longer and more precise problem descriptions starting from the seed of the initial query. A natural language parser is used both in the application of context-sensitive synonymy rules and the construction of the refinement tree.

2.1.7 Web Search

Joho and Jose 2006

Abstract [P2]

Presentation of search results in Web-based information retrieval (IR) systems has been dominated by a textual form of information such as the title, snippet, URL, and/or file type of retrieved documents. On the other hand, document's visual aspects such as the layout, colour scheme, or presence of images have been studied in a limited context with regard to their effectiveness of search result presentation. This paper presents a comparative evaluation of textual and visual forms of document summaries as the additional document surrogate in the search result presentation. In our study, a sentence-based summarisation technique was used to create a textual document summary, and the thumbnail image of web pages was used to represent a visual summary. The experimental results suggest that both have the cases where the additional elements contributed to a positive effect not only in users' relevance assessment but also in query re/formulation. The results also suggest that the two forms of document summary are likely to have different contexts to facilitate user's search experience. Therefore, our study calls for further research on adaptive models of IR systems to make use of their advantages in appropriate contexts.

Peng and Ounis 2007

Abstract [P28]

Query independent features (also called document priors), such as the number of incoming links to a document, its PageRank, or the length of its associated URL, have been explored to boost the retrieval effectiveness of Web Information Retrieval (IR) systems. The combination of such query independent features could further enhance the retrieval performance. However, most current combination approaches are based on heuristics, which ignore the possible dependence between the document priors. In this paper, we present a novel and robust method for combining document priors in a principled way. We use a conditional probability rule, which is derived from Kolmogorov's axioms. In particular, we investigate the retrieval performance attainable by our combination of priors' method, in comparison to the use of single priors and a heuristic prior combination method. Furthermore, we examine when and how document priors should be combined.

2.1.8 Structure/XML

Wang and Roelleke 2006

Abstract [P29]

Structured document retrieval requires the ranking of document elements. Previous approaches either aggregate term weights or retrieval status values, or propose alternatives to idf, for example, ief (inverse element frequency). We propose and investigate in this paper a new approach: Context-specific idf, which is, in contrast to aggregation based ranking functions, parameter-free.

Cornacchia and de Vries 2007

Abstract [P30]

This paper introduces the concept of a Parameterised Search System (PSS), which allows flexibility in user queries, and, more importantly, allows system engineers to easily define customised search strategies. Putting this idea into practise requires a carefully designed system architecture that supports a declarative abstraction language for the specification of search strategies. These specifications should stay as close as possible to the problem definition (i.e., the retrieval model to be used in the search application), abstracting away the details of the physical organisation of data and content. We show how extending an existing XML retrieval system with an abstraction mechanism based on array databases meets this requirement.

2.1.9 Multimedia

Chen *et al* 2006

Abstract [P31]

In this paper we consider episodic memory for system design in image retrieval. Time and location are the main factors in episodic memory, and these types of data were combined for image event clustering. We conducted a user studies to compare five image browsing systems using searching time and user satisfaction as criteria for success. Our results showed that the browser which clusters images based on time and location data combined was significantly better than four other more standard browsers. This suggests that episodic memory is potentially useful for improving personal image management.

Boyer and Brun 2007

Abstract [P32]

The identification of reliable and interesting items on Internet becomes more and more difficult and time consuming. This paper is a position paper describing our intended work in the framework of multimedia information retrieval by browsing techniques within web navigation. It relies on a usage-based indexing of resources: we ignore the nature, the content and the structure of resources. We describe a new approach taking advantage of the similarity between statistical modelling of language and document retrieval systems. A syntax of usage is computed that designs a Statistical Grammar of Usage (SGU). A SGU enables resources classification to perform a personalized navigation assistant tool. It relies both on collaborative filtering to compute virtual communities of users and classical statistical language models. The resulting SGU is a community dependent SGU.

Table 2: Overview of current research in IR domain

<i>Paper</i>	<i>Features</i>				
	Broad Topic Area	Model Based	Implemented	Operational	Evaluated
<i>Formal Models</i>					
Wang <i>et al</i> 2006 [P13]	-	+	+	+	+
McDonald 2007 [P14]	-	+	+	+	+
<i>Design and Evaluation</i>					
Demartini & Mizzaro 2006 [P15]	+	-	-	-	-
Mooney <i>et al</i> 2006 [P16]	+	-	-	-	-
Chernov <i>et al</i> 2007 [P17]	-	-	-	-	-
Yeung <i>et al</i> 2007 [P18]	-	+	+	+	+
<i>Topic Identification and News Retrieval</i>					
Yao <i>et al</i> 2006 [P19]	+	+	+	+	+
Shah & Eguchi 2007 [P20]	+	+	+	+	+
<i>User Interests and Workspaces</i>					
Bogers & v.d. Bosch 2006 [P21]	-	+	+	+	+
Vildjiounaite & Kallio 2007 [P22]	+	+	+-	-	-
<i>Clustering and Classification</i>					
Osinski 2006 [P23]	-	+	+	+	+
Sevillano <i>et al</i> 2007 [P24]	-	+	+	+	-
<i>Refinement and Feedback</i>					
Yamout <i>et al</i> 2006 [P25]	-	+	+	+	+
Roulland <i>et al</i> 2007 [P26]	-	+	+	+	+
<i>Web Search</i>					
Joho & Jose 2006 [P2]	-	+	+	+	+
Peng & Ounis 2007 [P28]	+	+	+	+	+
<i>Structure/XML</i>					
Wang & Roelleke 2006 [P29]	-	+	+	+	+
Cornacchia & de Vries 2007 [P30]	+	+	+	+	-
<i>Multimedia</i>					
Chen <i>et al</i> 2006 [P31]	-	+	+	+	+
Boyer and Brun 2007 [P32]	+	+	-	-	-

2.2 Discussion

The essence of the research assignment was to give an overview of the recent development within the Information Retrieval (IR) research domain. The papers explored are from the 2006, 28th or 2007, 29th European Conference on IR Research, these conferences are organized annually by several universities and companies. Within this research the papers were categorized within the following domains: Formal Models, Design and Evaluation, Topic Identification and News Retrieval, User Interests and Workspaces, Clustering and Classification, Refinement and Feedback, Web Search, Structure/XML and Multimedia. After being categorized they were reviewed based on several features namely: Broad Topic Area, Model Based, Implemented, Operational and Evaluated.

The work presented in several papers seem to be in more or less the same state of development, an example being McDonald 2007 [P14] and Yeung *et al* 2007 [P18] seeing that they both focus on a narrow topic area, the implementations are both model based and have both been evaluated. Another obvious observation is that the paper Chernov *et al* 2007 [P17] has neither of the proposed features, this because the authors only present a design for a desktop search dataset. Furthermore the work covered in Peng & Ounis 2007 [P28] is noticeable since it contains all the features, the reason for this paper covers a thorough research in the Web IR domain. And the work presented in Vildjiounaite &

Kallio 2007 [P22] is the only one that scored a “+ -” for a feature namely “Implemented”, this because this work was not fully implemented and that it was not operational yet.

Joho & Jose 2006 [P2] presented an interesting approach to web search result visualization, by presenting the user with optimized information within the search results, they conclude that this approach improves the search experience for the user. It assists the user in more rapidly finding target information. They also state that the main purpose of search engines is to help people find relevant information on specific tasks. Considering that finding relevant information is essential to be able to complete the task at hand, it is imperative that search interfaces are designed to optimize the amount of information users can obtain during a search. The results presented in Joho & Jose 2006 [P2], show that they have presented a means of improving the current search process.

It seemed interesting to verify if their model would also apply to the Enterprise search domain. They present a vague implementation and quite a vivid user evaluation model. This is why the following chapter focuses on further research on development Information Retrieval systems. From this further research the necessary knowledge is acquired to build the intended Enterprise Search solution. The user evaluation model presented by Joho & Jose 2006 [P2] is re-used and modified as presented in paragraph 7.4. By using their work as a basis and broadening the application within Enterprise Search, this research on search result presentation, contributes to science of Information Retrieval.

The following chapter focuses on the main elements that were researched in order to design and develop the new system of Enterprise Information Retrieval; most of these were taken from this initial research.

3 Selection from Related Work

This chapter focuses on the initial phase of the project, here research was done on related work and knowledge was gathered on the specifics of building an Enterprise Search Engine. First the additional search result visualization elements Top Ranking Sentences and Thumbnail are discussed. Then the research done on the anatomy of a search engine is presented. Next the following two paragraphs discuss Solr, an open source Search Engine and IBM Omnifind a Commercial Search Engine. The final paragraph discusses the research phase and justifies several decisions made after the research phase.

3.1 Search Result Elements

This paragraph discusses the search result visualisation elements Top Ranking Sentences and Thumbnail. These are textual and visual means of document summarisation that can be used for search result presentation purposes.

3.1.1 Top Ranking Sentences

There are several ways to create textual document summaries; one of them is used frequently in search engine interfaces namely a snippet. This element is basically a small piece of text containing part of the retrieved documents' text in which the users query terms reside; an example is shown in Figure 2. The yellow area is referred to as a snippet, in this case from Google search results.

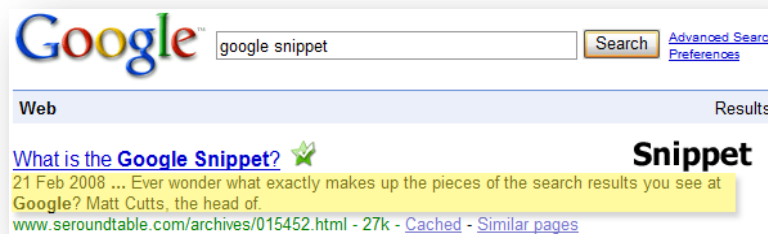


Figure 2: Google snippet example

Another way of creating a textual document summary is by using Top Ranking sentences (TRS). The definition of TRS as presented by White et al 2005 [P3] is “*TRS are the most potentially useful sentences in the top-ranked documents, extracted and scored according to factors such as their position in the source document, the words they contain, and the proportion of the query terms they contain*”. The factor “position in the source document” is relevant because the introductory sentences are important within a document, seeing these sentences usually give a nice overview of the document itself. While the factor “words they contain” can be a measure of how often do the query terms occur within the sentence; it is also possible to take into account whether the query terms within the document are embolded or if they are from the document title or headings. The last factor was “proportion of the query terms they contain” can easily be summarized as the amount of query terms appearing in the sentences, this component also ensures that the sentences retrieved are query relevant.

One of the main purposes of Top Ranking Sentences is to help searchers target potentially useful information. Potentially relevant sentences appear near the top of the list guiding searchers towards the answer they seek or documents of interest. These sentences encourage interaction with the content of the retrieved document set, an approach referred to in White et al 2005[P3] as “content driven

information seeking”. This is in contrast to query driven approaches where searchers proactively seek information through the query they provide. TRS have been well researched and that their usefulness has been demonstrated in several research papers some examples being Joho and Jose 2006 [P2], White et al 2005 [P3] and Tombros and Sanderson 1998 [P12]). But each of these papers study a different aspect of the use of TRS. The first paper uses these in a web environment, while the second one reports that query based sentences assists searchers, on making proper relevance assessments. Finally the last study used TRS as a replacement for the current snippet.

This research will use this textual document summary technique within the Enterprise Search environment.

3.1.2 Thumbnail

This paragraph gives an overview of the work done related to a visual document summary namely the Thumbnail. They are being used within most operating system’s file browser for users to quickly recognise the contents of the documents located in a folder, an example is given in Figure 3.

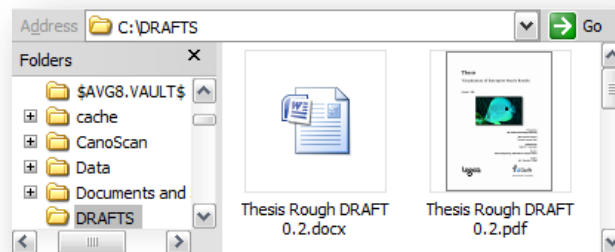


Figure 3: File Browser Thumbnail Example

In Figure 3 shows that the PDF document has a nice Thumbnail, while the Word document does not. When a user is looking for a certain document knowing the picture on its cover this thumbnail can actually help during this process, preventing the user from having to open each document. As mentioned earlier the Word document does not have a thumbnail, because it is not yet possible to render these from any file type. As can be imagined this is very easy when dealing with pictures because the picture just has to be resized. But as the document types get more complex it also gets harder to generate this simple form of a visual document summary.

The addition of this element to search results has not been implemented yet on a large scale, because clearly there is still need for further research. In the paper Dziadosz and Chandrasekar 2002 [P4], they demonstrate that users make better relevance decisions about web search results when a thumbnail is added to the interface.

3.1.3 TRS & Thumbnail Application

In this paragraph an application of both these textual and visual document summaries is briefly discussed. The paper Joho and Jose 2006 [P2] presents a comparative evaluation of textual and visual forms of document summaries as the additional document surrogate in the search result presentation. Their work explores the application of the TRS and Thumbnail in a Web Search environment. They develop and evaluate this system. It is not described in detail, the only information available is, that the system uses Google as a backend and that their system produces search results in the form shown in Figure 4. They also state that the system takes quite some time to calculate the TRS, but an exact figure

is not given. Their main focus is the evaluation of the system, which demonstrates that the addition of these elements assists users of during a Web Search Task. They also indicate that the system shows that TRS and Thumbnails have a positive influence on query reformulation and relevance assessment.



Figure 4: Search result presentation layout of system described in Joho and Jose 2006 [P2]

3.2 Anatomy of a Search Engine

This paragraph elucidates the inner workings of a large scale Search Engine, the basic components of a search engine are explained, and then Google's [P5, B2] approach is elucidated, because the basic components of both Web and Enterprise Search engines are similar and also because of its success and ingenuity. This was part of the initial research on how to design the basic components for the Basora Enterprise Search engine.

A Search Engine can be split up into the following basic components: a Crawler, a Parser, an Indexer and, a Searcher and a User Interface (Figure 5). These components function together to form a fully functional search application. The *Crawler* ensures that the new documents, web pages, images etc are all gathered for the *Parser* which is responsible for parsing the data. This means extracting all the metadata and actual document data for indexing. Then the indexer goes through all the parsed data and identifies all of the important keywords and builds a table that indicates where the terms are used within the document, web page, etc. Finally the index is used by the *Searcher* to answer the queries that are specified by the user.

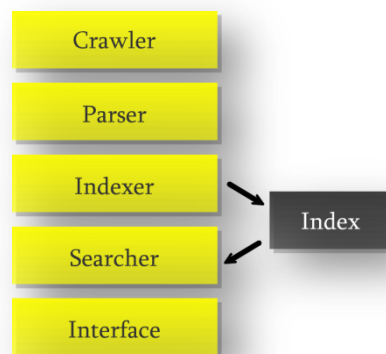


Figure 5: Basic search engine components

After this overview a brief review is presented of the Google search engine. The Google Search Engine can be split up into the following elements (as seen in Figure 6): Uniform Resource Locator (URL)

server, Crawler, Store Server, Indexer, Barrels, URL Resolver, Sorter and, Searcher, next all these elements will be portrayed.

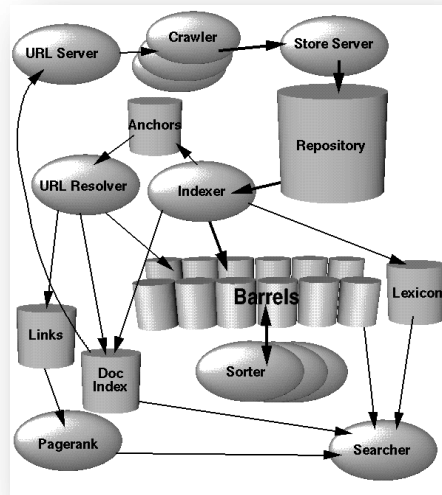


Figure 6: Components of the Google Search Engine (from Brin and Page 1998[P5])

The *URL Server* is responsible for sending new web page URL's to the crawlers. Now the *Crawler* fetches the sites that are fed to it through the *URL Server*. In addition the *Store Server* compresses and stores all the web pages into a repository. Furthermore the *Indexer* has to read the repository, uncompress the web pages and parse these. Afterwards the web pages are all converted into sets of word occurrences also referred to as Hits; these are stored in the *Barrels*. The indexer also parses and stores all the links from a web page in a so called anchor files. A *Hit* contains a word, its position within a web page, an approximation of the font size and its capitalization. Furthermore the *URL Resolver* reads the anchor files and converts the relative URL's into absolute ones and assigns every page a document identifier. It also generates a database of links which are pairs of document identifiers. This links database is used to calculate the PageRanks for all the pages. These *PageRanks* are used to determine the relevance or importance of a web page. The *Sorter* re-sorts the contents of the barrels by word identifier to generate an inverted word index. Another feature of the *Sorter* is that it produces a list of word identifiers and offsets into the inverted index. A program so called DumpLexicon takes this list and together with the lexicon produced by the indexer; it generates a new lexicon to be used by the searcher. The *Searcher* answers queries by using the lexicon, together with the inverted index and the *PageRanks*. From this research the knowledge needed of basic components of Search Engine's was extracted. It was also thought that some of the ingenuity of this design could be used in the construction of the Basora Enterprise Search prototype.

3.3 Solr 1.2 Search Server

Solr¹ is an open source Enterprise Search server based on the Apache Lucene² Java Search library. Apache Lucene is a high-performance, full-featured text search engine library written entirely in Java,

¹ <http://lucene.apache.org/solr/> last visited: 20 October 2008

² <http://lucene.apache.org/> last visited: 20 October 2008

developed by the Apache Software Foundation³. The Apache Software Foundation provides support for the Apache open source software projects; these are characterized by a collaborative, consensus based development process, and an open software license. They also strive to create high quality software that leads the way in its field.

First some history on Solr⁴, it was first developed at CNET Networks and was donated to the Apache Software Foundation in early 2006 under the Lucene top-level project umbrella. During its incubation period, Solr steadily accumulated features and attracted a robust community of users, contributors and committers. Now after its incubation, Solr is a subproject of Apache Lucene.

Solr uses the Lucene Java Search Engine library to provide a user with a power full Enterprise worthy Indexer and Searcher. As previously shown in Figure 5 a Search Engine consists of the components Crawler, Parser, Indexer, Searcher and User Interface. The Solr Search server only consists of an Indexer and a Searcher as shown in Figure 7.

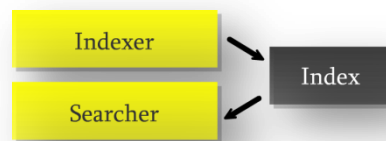


Figure 7: Solr components

These components are responsible for allowing a system to quickly retrieve the results, once a user submits a query. Customization of these components can be done quite easily depending on its application. Most of the customization can be done through editing several XML configuration files, but if a user wishes to modify the entire search server (Indexer, Searcher); this can also be done by developing these additions in Java, onto the existing framework.

This project is well documented using Javadoc and there is also a Solr Wiki⁵ available which explains the Solr components and uses quite vividly. It is one of the most well known and robust open source Search servers available.

3.4 IBM Omnifind Search Solution

There are several commercial Enterprise Search Solutions available namely, IBM Omnifind, Google Enterprise, Fast, etc. Within Logica IBM Omnifind is a widely used Search Solution, which is available for use by its employees, both for developers and users. Logica developers have implemented this solution for both themselves and several clients.

The main goal of IBM Omnifind is to provide scalable, secure, high-quality enterprise search⁶. It is designed to improve the productivity of knowledge workers and maximize the value of a company's portal and collaboration investments. Its main features include:

- Scalable to millions of documents and thousands of users
- Pre-build integrations for indexing data and content from file shares, databases, collaboration tools, systems, blogs, etc.
- Built using the Java programming language

³ <http://www.apache.org/> last visited: 20 October 2008

⁴ <http://www.ibm.com/developerworks/java/library/j-solr1/> last visited: 20 October 2008

⁵ <http://wiki.apache.org/solr/FrontPage> last visited: 20 October 2008

⁶ <http://www-01.ibm.com/software/data/enterprise-search/omnifind-enterprise/> last visited: 20 October 2008

- Uses Servlet technology.

This Enterprise Search solution is very large, robust and commercialized. Its interface is shown in Figure 8, this is used to query the search engine and obtain and analyse their search results. The search result presentation layout of the IBM Omnifind 8.5 Enterprise Search Solution is shown in Figure 9.

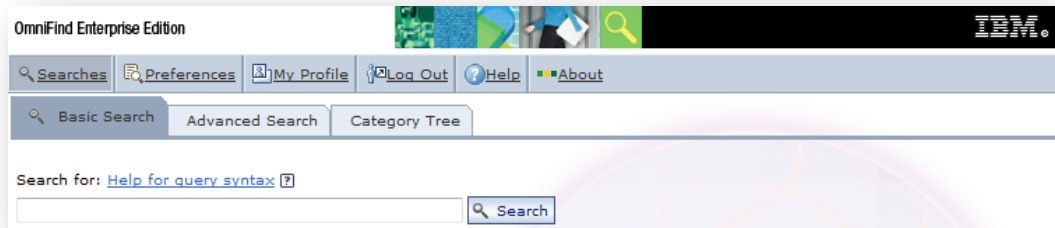


Figure 8: IBM Omnifind 8.5 Enterprise Search Interface

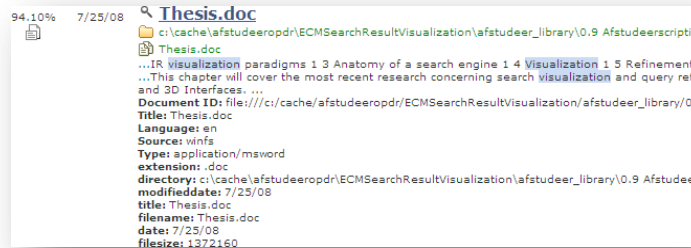


Figure 9: IBM Omnifind Result presentation

The IBM Omnifind Search Solution provides companies with a search engine that performs adequately within an Enterprise environment. For developers IBM provides, several forms of support including a help and support page, and several manuals for installation and customization. They also provide personal support by their team of highly trained professionals.

3.5 Discussion

During the research phase of this project, several aspects required for the development of the Basora Enterprise Search prototype were explored, namely user interface components (Top Ranking Sentences, Thumbnails), and Search Engine components (Crawlers, Parsers, Indexers and Searchers). These user interface components were chosen because of the positive results of several studies using one of these or both of these components (Joho and Jose 2006[P2], White et al 2005 [P3], Tombros and Sanderson 1998[P12] and Dziadosz and Chandrasekar 2002 [P4]).

Google Search Engine is researched because; the basic components of both Web and Enterprise Search engines are similar, and also because maybe some of their basic concepts might be used during the development of the Basora Enterprise Search prototype.

Finally the Solr 1.2 Search server and IBM Omnifind Search Solution were explored because of their Indexer and Searcher components. Seeing that the main focus of this research was not to implement

these components, and since an implementation already exists and which is well tested, the development of the Indexer and Searcher seemed unnecessary. Solr and IBM Omnifind were researched to be used for these components of the Basora Enterprise Search prototype.

The Solr Search server was chosen because it is an open source project and could be manipulated to provide all the functionality needed for the prototype; while the IBM Omnifind Solution was less adjustable to the Basora prototype development requirements. It was also very difficult to access the Indexer and Searcher components of the IBM Omnifind Solution for integration into the Basora prototype. Another key factor was that the Solr server, was well documented and tested; the documentation was easily accessible, unlike the documentation for the IBM Omnifind Solution. This was very hard to obtain, often it was incomplete or too superficial. It was possible to contact the IBM support line, but this was usually quite a lengthy process. In the end Solr 1.2 was chosen because of its speed, agility and good documentation.

4 Cognitive Model of IR

The main focus of this thesis is the search of documents from a file system, within an Enterprise environment. Within these Enterprise file systems various file types are found namely, Microsoft Word, PowerPoint, Excel, Access, PDF, XML, Plain Text, etc. This research only focuses on the retrieval of Microsoft Word, PowerPoint, Excel and PDF documents.

Within the chapters on initial research was concluded that the search process within Web and Enterprise environment's are very similar. Most users are familiar with Google search which is a full text search engine for the Web environment; which is very similar to an Enterprise Search Solution. This is supposed to offer a user the same functionality but instead allowing a user to find documents housed within a company. In order to obtain a good overview of the cognitive model of IR within a company, several relevant users were observed within their working environment and also interviewed (Secretaries, Project Managers, Architects, etc.). From this two types of users were distinguished an experienced user and an inexperienced user. An experienced user knows exactly how to use a system, how to define a search query, how to interpret the search results and also how to refine their search query. Finally an inexperienced user is not aware of the system and the search procedure and need to be guided through this process. The following paragraph presents a cognitive search model.

4.1 Search Model

The following steps are considered in the search process cycle namely: A. User defines a search query, B. the system searches for results, C. the system displays the search results and D. the user interprets the search results and refines his query. The steps A and D are related to the user ontology, while the steps B and C are part of the system ontology. This search process cycle is illustrated in Figure 10. Now for each of these steps a paragraph is dedicated to elaborate on these.

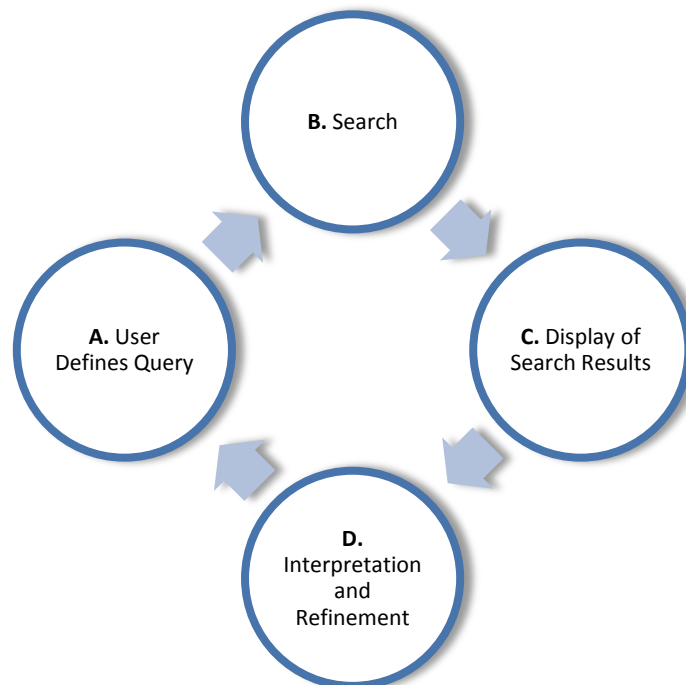


Figure 10: Cognitive Model: Search Cycle

4.1.1 A. User Defines a Query

Most search systems based on full text search, use keywords as their main form of input. Some users also like to add features like the date the document was created, the size of the document, name of the author, department where the author works, etc. to their search query. Most of the keywords that a user provides are related to the topic on which their search is based.

We cannot assume that an inexperienced user is aware of the key terms that are used by the system, so a different ontology is required for the user and for the system. This indicates that there is a need for an ontology mapping, ensuring that an inexperienced user's keywords are also understood by the system. This can be achieved by first allowing the system to check if the keywords are available in the systems ontology, if not mapping the keywords from user to system ontology by using WordNet⁷ for relevant terms. The system can also define a set of frequently used search terms (predictive search) and allow the user to refine his search from here out. These terms can be obtained by observing a users search behavior and history of search terms to define these predicted keywords.

An experienced user usually knows exactly what he wants, so this search process is quite straight forward. They fill in the terms usually, title of the document, name of the author and other exact key terms and locate their desired document. Now other users only have a vague idea of what they are searching for making the search process more difficult. Some systems use a predefined set of keywords to facilitate this user during his search, only requiring him to score these keywords. But when developing the new system for Enterprise Information Retrieval a more bottom up approach is preferred. The user is allowed to define his own keywords, using his own language, this fits nicely in the paradigm user centered approach "bottom up" [B1].

4.1.2 B. Search

For the searching step of the search cycle, several implementations were considered namely the IBM Omnifind Enterprise Edition Search System and the Apache Solr Search Server. As discussed in paragraph 3.5, the Apache Solr Search Server was chosen as the backend of the new Enterprise Search system. It was chosen because of its open-source nature and documentation. Its actual search algorithm is quite fuzzy, seeing it allows for typos. An example being when searching for the term "Boat" it also considers words like "Boats", "Boot", "Boots", etc. and if there are very little results it also considers sub words like "Boa" and such. The Solr Searcher uses a Porter Stemming Algorithm⁸ to remove the commoner morphological and inflexional endings from words in English. This is mainly used as a part of term normalisation that is usually done when setting up Information Retrieval system. The Apache Solr Server is also discussed in paragraph 3.3.

4.1.3 C. Display of Search Results

There are various ways of presenting search results, some of these are discussed in paragraph 3.1. It is possible to show a user a fragment of text from a document that contains the query terms used, this is otherwise known as a snippet (discussed in paragraph 3.1.1). It is also possible to show a user an automatically generated summary of a document [P33, P34]. But there is no algorithm available yet to do this in real time. So for this new system, Top Ranking Sentences and a Thumbnail were chosen as the document summaries as discussed in paragraph 3.1. These summaries have been well researched

⁷ WordNet is a large Lexical database of English. <http://wordnet.princeton.edu/> last visited 20 October 2008

⁸ Porters Stemming Algorithm <http://tartarus.org/~martin/PorterStemmer/> last visited 20 October 2008

and several papers show that the addition of these elements actually helps a user with query reformulation and relevance assessment [P2, P3, P4] (discussed in paragraph 3.1).

4.1.4 D. Interpretation and Refinement

During the interpretation and refinement phase it is possible that the user is confronted with a large set of probably relevant documents, usually because the query that was defined was vague or that there are a lot of similar documents on the requested topic. From a user friendly aspect the system can order these documents based on relevance. Another important aspect is that the user should be able to immediately see what documents are relevant and which ones are not, from the elements presented in the search results. Some users are more text oriented, while others are more visually oriented. For both these type of users the new system offers a solution.

- Instead of just implementing an existing *enterprise* [55]
- Thesis Visualization of *Enterprise* Search Results October 3, 2008 Programme MSc. [25]
- Gregory Wall? mauricewalle@gmail.com Visualization of *Enterprise* Search Results Thesis Report Friday, [25]

Figure 11: Textual Document Summary - Top Ranking Sentences

For the text-oriented users the new system offers the addition of the textual document summary TRS as introduced in paragraph 3.1.1. These sentences allow a user to formulate a better understanding of the contents of a document. Furthermore the addition of a Thumbnail to the search result presentation is intended to facilitate the searching process for the more visually oriented searchers. This document summary allows a user to quickly identify key features of the front page of document.



Figure 12: Visual Document Summary - Thumbnail

A user is familiar to a document will recognise this immediately, through the images on the first page, design of the front page and the ability to read the large fonts. Both these document summaries are complementary of each other and fit nicely within the human perception process [B1].

In case that a user cannot find the intended document from the search result because the result list is too large or that it does not contain the desired document. The user needs to reformulate his query in order to better indicate to the system, what he is looking for. This can be done in various ways one way is for the user to either type new query terms or add or remove some query terms. It is also possible that the user indicates what documents are relevant to his search and which ones are not. So the search engine can use these documents, as an indication of what the user is looking for (This has not been implemented in the current system). An example of such a system is the “Aquabrowser”⁹.

It is also possible to set up a user profile by either allowing a user to define this beforehand or use data-mining techniques to automatically generate user profiles. These profiles can also be used when

⁹ <http://www.aquabrowser.com/> last visited 20th October 2008

refining the search results, by using the information of what a user searches for often to offer the user some predicted query terms that may be used to refine the search.

The new system developed does not support all the features discussed above but it does allow a user to start with a very vague query and incrementally refine this until he finds his target document. For a good overview of the features available in the Basora Enterprise Search system please see chapter 6.

4.1.5 Search Cycle Examples

This paragraph gives several examples of user's going through the search cycle discussed in the previous paragraphs. The first example is of an inexperienced user, which needs a lot of iterations through the search cycle. Now the second example is of a moderate user that is quite proficient at using a search system. It requires the moderate user less iterations than the inexperienced user to find his target document. And the last example is of an experienced user that actually finds his document in a single iteration.

Example 1: Inexperienced User

In this paragraph an example is given of a new intern within a company, which is looking for information on his assignment. The only information available to him is the topic of his research which is real-time business intelligence. So he opens up his browser and goes to the company's trusted Enterprise Search Solution.

A. User defines a query

Seeing that the user only has the name of his topic to go on, he types in "real-time business intelligence" as his first query, and submits this to the system.

B. Search

The system now receives the query, converts the user ontology terms to system ontology terms. Next the system goes through its index to locate all documents that are relevant to this query.

C. Display of search results

Now the system extracts all the necessary document summaries and relevant information to format these into the search result list. Once this is completed it presents them to the intern.

D. Interpretation and refinement

The intern gets a list of over 10.000 documents back from the system, and the first 10 are displayed. He starts going through these and finds that none of these are related to his research topic. Then he remembers that the department that deals with this topic is called IDT. So he adds this to his query terms and resubmits his query. Again the system processes this and displays the search results, steps B and C. Now the intern again gets a list of results but this time only half the amount of documents are returned. So he starts going through the list and discovers a document that seems interesting, but it is not exactly what he is looking for. So the intern decides to add the name of the author of that document to his search terms, and he submits the query again to the system. Steps B and C are carried out by the system again and now the intern is presented with yet another list of documents. But this list is very short only 10 documents, he starts going through these clicking on each of them one by one reading their contents. Finally he finds a document that he can use for his research.

Example 2: Moderate User

In this paragraph an example is given of a secretary carrying out a search task. Let's say the secretary is looking for a payroll document of last month of the department Human Resources. So she opens up her browser and goes to the company's trusted Enterprise Search Solution.

A. User defines a query

She first thinks about what query she is going to use to locate this document. She knows the document contains the words payroll, human resources. She also knows that the document was created last month. So she types in "payroll human resources" and submits her query.

B. Search

The system now receives the query, converts the user ontology terms to system ontology terms. Next the system goes through its index to locate all documents that are relevant to this query.

C. Display of search results

Now the system extracts all the necessary document summaries and relevant information to format these into the search result list. Once this is completed it presents them to the secretary.

D. Interpretation and refinement

The secretary now looks through the results and sees that there are many results but none of these are exactly what she is looking for. So she tries to refine the search by also adding last month's date to the query term and re-submits her search. This takes her back to step A in the search cycle, then the system executes step B and D. Finally she looks through the search results and finds the document she is looking for.

Example 3: Experienced User

In this paragraph an example is given of a project manager that is locating a document that he is familiar with. He is looking for a thesis of one of the interns of which he remembers the intern's name, his project title and the date that the intern graduated. So he opens up his browser and goes to the company's trusted Enterprise Search Solution.

A. User defines a query

He starts formulating a query containing the intern's name, project title and the word "thesis" submits this to the search system.

B. Search

The system now receives the query, converts the user ontology terms to system ontology terms. Next the system goes through its index to locate all documents that are relevant to this query.

C. Display of search results

Now the system extracts all the necessary document summaries and relevant information to format these into the search result list. Once this is completed it presents them to the manager.

D. Interpretation and refinement

Now the manager knowing the date that the intern graduated also filters the results using this date as the filtering criteria. Now the system returns only 4 results each of these are a version of the document he is looking for. He locates the last version and opens this to continue carrying out the task at hand.

5 Global Design

After discussing the initial Research Phase of this project, this chapter gives an overview of the design the Basora Enterprise Search prototype. The knowledge gathered during the Research Phase was essential in order to design the prototype; this was used to make critical design decisions.

First the methodology used during the design and development of the prototype is elucidated. Then the global design of the Basora Enterprise Search (BES) prototype is presented; here the crawling, parsing, committing, indexing, searching and interface components are discussed. Hereafter the custom TRS algorithm used for the TRS Ranking calculation is discussed.

5.1 Methodology

As mentioned in paragraph 1.4, an agile approach was used during the design and development of the prototype; this is conform to an incremental approach. First an initial design was created for the entire system and this was broken down into requirements. Then a planning was created as to which requirements needed to be completed and when. Over a time-span of 12 weeks, every 3 weeks a meeting was planned with the stakeholders at Logica, so they could see how the prototype evolved and so they could give input. This input usually consisted of minor additions or modifications. For each of these sessions a document (Example of these documents is available in Appendix A: Update Meeting Sheet) was created and given to the stakeholders before the meeting. This document contained an overview of the requirement that were completed, along with the ones that were not. Additionally it contained the complete planning; this consisted of a Gantt chart, and the actual activities that still needed to be carried out. After the meeting the additional requirements list was updated, in order of importance. So the important requirements were implemented first.

At the end of the Design and Development Phase a meeting was held with the stakeholders from Logica so, the prototype could be assessed and discussed, to verify that the prototype performed to their satisfaction. For this meeting as well a document was created beforehand (available in Appendix B: Final Assessment Prototype), containing all the requirements that had been implemented, along with the planning. Additionally it also contained a list of future additions. At the end of this process a functional prototype was created of an Enterprise Search Engine with an improved search result presentation technique.

5.2 Basora Enterprise Search Global Design

The final design of the Basora Enterprise Search prototype can be split up into the following components namely Crawler, Cacher/Parser, Committer, Indexer & Searcher, and Basora Engine & Web Interface (as seen in Figure 13). Each of these components performs a vital role in order to allow the prototype to function. The Crawling, Parsing and Committing components are also referred to as the Basora Pre-processing stage. It has to be done before-hand for the prototype to work. The following paragraphs elaborate on the role each of these fore fills and also on the capabilities of these components. All the components were developed in Java 1.6, because of its platform independence and the availability of the PDFBox and Apache POI Java libraries; which are used to read the supported file formats.

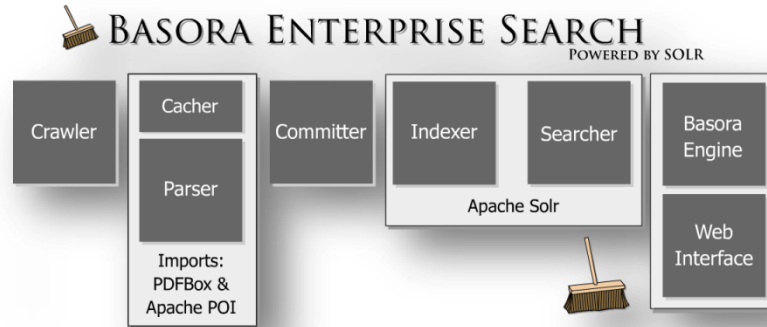


Figure 13: Basora Enterprise Search prototype components

5.2.1 Basora Pre-Processing

The components Crawler, Cacher & Parser and Committer can be referred to as the BES prototype's pre-processing step. These steps need to be undertaken before the system can be used. The following paragraphs discuss the design of the pre-processing components.

Crawler

The crawler component of the Basora Enterprise Search system is able to crawl a file system and retrieve the location of all the, Microsoft Word (.DOC), Microsoft PowerPoint (.PPT), Microsoft Excel (.XLS) and PDF documents (.PDF), along with the date and time it was last modified and it should also assign each document a unique identifier. This is used when the crawler re-crawls the file system it can identify whether it has already crawled the file and also it can check if the file has changed in between crawling sessions. Word (DOCX), Excel (XLSX) and PowerPoint (PPTX) 2007 documents are not crawled because the Parser does not support these file types this is further explained in paragraph 0. After crawling the file system the crawler stores the information of the files crawled into crawl-files (.crawl). Each of these contains a maximum of 100 document-locations, etc, in order to keep the time it takes to read the files into the system to a minimum. These crawl-files are stored in a folder to be fed to the parser. The crawler was developed from scratch completely using the Java 1.6 standard library, no additional libraries were needed; further implementation details are available in paragraph 6.2.1.

Parser

Now the Parser component is responsible for reading through the documents and extracting all the metadata and text. This is formatted into XML, so it can be committed to the Solr Indexer, which only supports XML files (explained in detail in paragraph 5.2.2). The parser first caches some of the files the crawler has retrieved to the local disk, this is done to reduce the parsing time due to network lag. Then it reads the metadata and contents of each of the files and then saves this onto the disk in XML format. This is repeated until all the crawled files have been parsed and converted to committable XML.

The Parser supports the file types: Microsoft Word (.DOC), Microsoft PowerPoint (.PPT), Microsoft Excel (.XLS) and PDF documents (.PDF). While the file types: Microsoft Word (DOCX), Microsoft Excel (XLSX) and Microsoft PowerPoint (PPTX) 2007 are not supported. Because during the development phase, an analysis was made of the fixed dataset that needed to be made searchable; from this it became clear that only 2% consisted of these file types (can be seen in paragraph 7.2). Additionally the Apache POI library (see paragraph 6.1.1 for details) used to read the Microsoft file types, did not have support for reading the Office 2007 files, and no other libraries could be found at the time. With this in mind the stakeholders decided that this was not a priority.

The Java libraries used to develop the parser are: Apache POI and PDFBox (see paragraph 6.1.1 for details). The Apache POI library contains functions to be able to read and write Microsoft Office documents, while the PDFBox library supports reading and writing of PDF documents. The parser itself was written from scratch in Java 1.6 using the libraries listed to read the contents of the files. The process of creating the committable XML files was also developed from scratch, using the standard Java libraries, the exact implementation details are discussed in paragraph 6.2.1.

Committer

The Committer is the link between the Basora Parser and the Solr Indexer; it is responsible for sending the Parsed XML files to the Solr Indexer. It reads the parsed XML files one by one and sends them to the Solr Indexer. This component was also developed in Java 1.6 and uses several posting functions readily available in the Solr Java library; this component is further discussed in paragraph 6.2.1.

5.2.2 Solr Indexer & Searcher

This component of the Basora Enterprise Search prototype has to index the XML files committed so these are properly stored and cached so once a search query is submitted to the Searcher it can quickly retrieve the relevant documents. These results are returned in XML form and still need to be processed by the Basora Engine before they can be presented to the user (this is explained in paragraph 5.2.3).

This part of the Basora Enterprise Search prototype has been configured using the Solr Indexer and Searcher. These components were used because developing these is outside of the scope of this research and that it's already available and that Solr is a well respected and tested Search Server. The configuration details are listed in the implementation chapter in paragraph 6.2.2.

5.2.3 Basora Engine

As explained in paragraph 5.2.2 the results are returned by the Solr Searcher in XML format, this XML contains all the necessary data to be able to extract the textual document summary Top Ranking Sentences. The TRS are extracted in real-time; this is part of the responsibility of the Basora Engine. Its main purpose is to function as a bridge between the Solr Searcher and the Basora Web Interface. A submitted query first goes through the Basora Engine where it is formatted and sent to the Solr Searcher, and then the results it obtains are parsed and formatted so this can be displayed to the user through the Web Interface.

This was developed from scratch using Java 1.6 and its standard libraries. A custom built XML parsing library was used for all the XML parsing purposes, this was done because this performs a factor 10 faster than the Java standard XML parser. Additionally the Basora Engine also ensures that a user can sort and filter the search results obtained. Furthermore the Engine also calculates the response time of both the Solr Searcher and the Basora Engine itself; the implementation details are available in paragraph 6.2.3.

5.2.4 Web Interface

The main interface consists of a basic search interface with a standard search query input box and a search button. There is a top banner containing the name of the prototype while the footer contains the logos of Logica and the TU-Delft; as shown in Figure 14. This design is based on the IBM Omnifind interface shown in Figure 8.

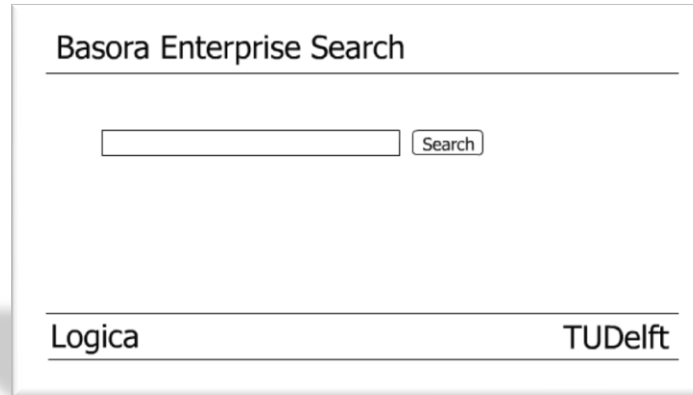


Figure 14: Basora main interface initial drawing

Now the visual and textual document summaries namely the Top Ranking Sentences and Thumbnail are placed in the interface as illustrated in the figure below. Four different layouts have been developed in order to properly evaluate the effectiveness of these. Layout 1 is a basic layout which has the components of current Enterprise Search engines interfaces, namely the filename, author, location and snippet. The basic metadata block (illustrated in Figure 15 to Figure 18) contains the variables: Relevance, Date last modified, Date created, Date indexed, File type and File size. While the second layout is equal to the baseline, with the additions of the textual document summary: Top Ranking Sentences (TRS). Furthermore the third layout is again equal to the baseline with the addition of a visual document summary. The last layout is a combination of the two previous ones. It also has the basic components of the baseline but also contains both the TRS and Thumbnail. In the figures below a schematic overview of the four layouts is presented.



Figure 15: Concept Layout 1

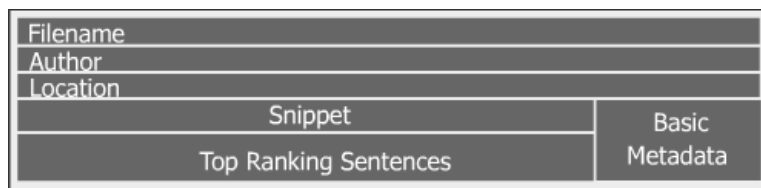


Figure 16: Concept Layout 2



Figure 17: Concept Layout 3



Figure 18: Concept Layout 4

The design of these layouts is based mutually on the current IBM Omnifind interface (shown in paragraph 3.4 Figure 9) and the design presented in the paper Joho and Jose 2006 [P2] (shown in Figure 4).

There is also a matter of choosing the dimensions of the Thumbnail this was done by first looking at the dimensions used in other work related to this and here they used the dimensions: 112 by 82 [P2]. In this work they indicated that this size was a bit small to read the text but large enough to grasp the visual aspects of the pages. So some example thumbnails were created with these dimensions and some with dimensions 200 by 150 as well. The latter was chosen because this size seemed to allow readability of the large fonts on the page, giving the user some extra information to work with.

The following paragraph gives the design of the TRS calculation algorithm that will be used for calculation of the ranking of the Top Ranking Sentences.

5.2.5 TRS Algorithm

This paragraph explains the Top Ranking Sentence extraction algorithm that was developed to calculate the TRS rankings. The variables were chosen based on the work described in the paper White et al [P3]. Seeing that during the research phase no actual algorithm could be found, this needed to be custom made, inspired by the literature available. The system extracts all sentences in which the query terms occur and ranks all these sentences accordingly. The three characteristics used are: position of a sentence within a document, total term occurrence, and amount of terms the sentence contains.

Sentence Position

Position of the sentence within the document is determined based on its location within the document. A sentence that occurs in the upper 20% of the document is usually part of either the table of contents, the introduction, or the executive summary, so this sentence has quite a high ranking namely 50 seeing that these are part of the document that are read most often and are usually very important. The lower 20% usually contains the summary or conclusion of a document which is also an important part of the document, so if the sentence is located here it will also get a high ranking namely 40. If the sentence is located elsewhere in the document it will be given a ranking of 10. The proportions 20 by 60 by 20 were obtained by gathering about 20 documents from the available dataset (presented in paragraph 7.2) and going through these and calculating the proportions; while the rankings 50, 10 and 40 were obtained while during the development phase by starting out by choosing several queries; then executing them with various ranking values and tweaking these (this process is described further later on in this paragraph).

Term Occurrence

Term occurrence is ranked as follows, first the total occurrence of query terms within all sentences are calculated and also the total occurrence of all these query terms within the sentence. Then the total

query term occurrence within a sentence is divided by the total query term occurrence over all sentences, this will be multiplied by 100 and rounded to integers.

Amount of Terms per Sentence

Amount of terms per sentence ranking will only be calculated if the user submits two or more query terms to the engine. If this requirement holds, the engine will calculate the amount of query terms within a sentence and divide this by the total amount of query terms supplied by the user, and multiplies this result by 100 and rounds this also to an integer.

Once these individual rankings have been calculated the engine calculates the total ranking per sentence based on Equation 1. The equation treats the Position Ranking as the variable with the most influence over the total ranking and the Amount of Terms per Sentence Ranking as the variable with the least. This approach is taken because this characteristic is described in the literature as quite an influential one [P3]. The exact values for the total ranking calculation were obtained by iteratively tweaking these. Once the entire algorithm was implemented it was thoroughly tested to evaluate whether the ranking function actually ranked more relevant sentences higher than less relevant ones. This was done by first choosing five fixed query terms and analysing the sentences and respective ranking obtained. The values were obtained by tweaking the values of the Total Ranking Calculation and the Sentence Position variable until the desired results were obtained. These calculations are done when returning the results to the user, so in real-time. This TRS algorithm was implemented in Java 1.6 using the standard Java libraries.

$\frac{(5x + 3y + 2z)}{10}$	x = Position Ranking y = Term Occurrence Ranking z = Amount of terms Ranking
-----------------------------	--

Equation 1: Total Ranking Calculation

6 Implementation

In this chapter the implementation phase of this project is described, the prototype is called Basora Enterprise Search. As explained in the introductory paragraph 1.2, Basora is a word in the language Papiamentu which means broom. In the following paragraphs; first the hardware, software, tools and libraries that were used are discussed. Then the implementation itself is elaborated on, the pre-processing steps, the interfaces and the ranking algorithms are discussed.

6.1 Hardware, Software, Tools and Libraries

This paragraph gives an overview of the hardware and software that was used during the development of the prototype. First the Operating System, Hardware, Software and Tools and Libraries needed for the development of the Basora Enterprise search engine is discussed. Hereafter an overview is given of the minimal PC requirements of the client/user side of the Basora Enterprise Search engine.

6.1.1 Server Side

The server end of the Basora enterprise search engine was developed to work on a machine with at least the following specification: Intel Core 2 Duo T7500 2.20 GHz (2 CPU's), 800 MHz FSB, 2 GB of RAM, 100 GB Hard disk drive, running Windows XP Professional Edition SP2 as the operating system. It should function though on any operating system seeing the system is developed using Java 1.6, which is platform independent. It also requires Apache Tomcat 5.5, as the server environment, the Basora Enterprise Search Engine prototype itself is a Java 1.6 Servlet based application. The prototype was developed in Eclipse EE, a Java Integrated Development Environment (IDE) with support for Java Servlet's and Java server pages. In the table below a small overview is given of the java packages that were used for the development of the prototype.

Table 3: Java packages used for development of Basora Enterprise Search Engine

Packages	Description
Apache Solr	<i>Apache Solr is an open source enterprise search server based on the Lucene Java search library</i>
Apache POI	<i>Apache POI is a library that can be used for reading and writing of Microsoft Word, Excel, PowerPoint files. It also supports thumbnail generation for PowerPoint files.</i>
PDFBox	<i>PDFBox is a java library that can be used to read and write PDF documents. It also supports thumbnail creation of PDF documents</i>

Apache Solr is an open source enterprise search server based on the Lucene Java Search library. It contains an indexer and a searcher, XML files need to be sent to the Solr server for indexing and after sending a search query to the server the results are returned in XML format.

Apache POI is a Java library that has support for writing and reading Microsoft Word (.DOC), Excel (.XLS) and PowerPoint (.PPT) files. The new office files standard is not yet supported namely: DOCX, XLSX and PPTX (as pointed out in paragraph 5.2.1). It is also possible to generate thumbnails of PowerPoint slides using this Java library.

PDFBox is a Java library with support for reading and writing PDF documents. Its latest version was not completely capable of rendering thumbnails of the document pages. Most functions were

implemented but it was not all functioning, so the library was extended in order for it all to work properly for the prototype. HTML 4.01, CSS 2.0 was used for the development of the browser based user interface; furthermore Adobe Flash CS3 was used to create all graphics needed; some examples of the graphics being the broom logo and the Basora banner. The following paragraph gives an overview of the minimal PC requirements for users of the BES system.

6.1.2 Client Side

For the client end of the Basora Enterprise Search prototype to function to its maximal potential the minimal requirements for the user pc should be: Intel or AMD processor 2 GHz, 1 Gigabyte of RAM and 64MB Graphics Card. Seeing that it is a web-based application no minimal required hard disk space is listed. The user interface is developed to work with Internet Explorer 7; the following browsers are also supported namely: Internet Explorer 6 & 8, Google Chrome and Firefox 2.0 & 3.0.

6.2 Basora Enterprise Search

This paragraph focuses on the development process of the Basora Enterprise Search (BES) prototype. As stated in paragraph 5.2 and shown in Figure 13 the BES prototype consists of the components: Crawler, Cacher/Parser, Committer, Indexer & Searcher, and Basora Engine & Web Interface. The implementation details of these components are presented in the following paragraphs.

6.2.1 Basora Pre-Processing

As discussed in paragraph 5.2.1 the Crawler, Parser and Committer components are all part of the BES systems pre-processing step. The following paragraph discuss the implementation of these components.

Crawling

The crawler that was implemented is capable of crawling any file system, whether it is a physical one or is network based. It crawls the file system for the files with extension: doc, ppt, xls and pdf (as previously discussed in paragraph 5.2.1). It registers the location of the file, its last modified date and it assigns the file a unique identifier, this is saved in a “CrawledFile” object (class diagram available in Figure 19). It does this in order to be able to re-crawl the system and only retrieve the new and changed files. In Figure 21 the class diagram of the crawler is given, it consists of the classes “Crawler”, “CrawlerController”, “CrawlerFileNameFilter” and “CrawlerMain”. The “Crawler” class is the main component of the crawler; in order to start a crawl the following steps should be taken:

- Instantiate the “CrawlerController” class
 - Its constructor automatically specifies the files to be crawled (“doc”, “pdf”, “ppt”, “xls”)
- Specify the directories that need to be crawled
 - This is done by using the “addCrawlPath(String directory)” method
- Run the crawler
 - Just execute the “crawl()” method.

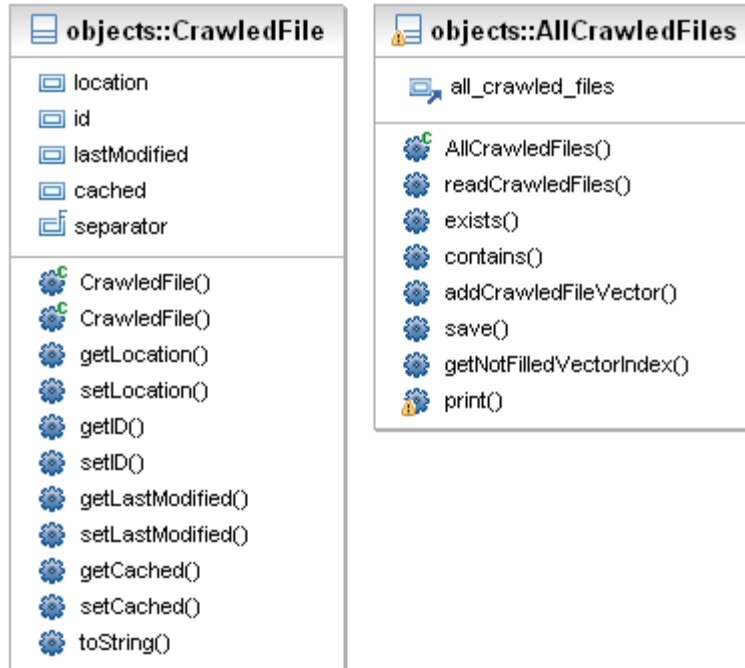


Figure 19: Crawler objects

The crawler can be run in its own thread as well by using the “crawl(true)” method instead of the “crawl()” one. While the crawler is running it will save all the “CrawledFile” objects to a file in batches of 100, this is done to ensure that the saved files do not get too big. The “AllCrawledFiles” object/class is used to keep track of all the files that have already been crawled so they are not crawled twice. The files that have already been crawled are read into this object every time the crawler is run. The method that actually goes through the file system and extracts all the documents is the recursive method listFolder(java.io.File), the general outline of the method is presented in Code Box 1.

Crawling Example

The crawling algorithm works as shown in Equation 2. Let’s use Figure 20 as an example file system; first the algorithm retrieves the folders A, B and C and the files that are located in the Root folder. These are then checked, if it is a folder the algorithm will execute step 2 using the given folder. Otherwise if it is a document that has the extension PDF, DOC, PPT or XLS, the document is given a unique identifier and its location and date last modified is stored (as discussed in the previous paragraph). This is recursively repeated until all the subfolders have been crawled.

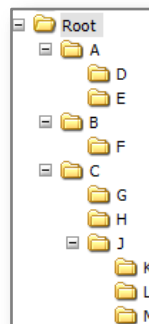


Figure 20: File system

```

1. Start at Root
2. foreach (file and folder)
3.   if ( is File )
4.     Check if it is a valid file format and crawl it
5.   else if ( is Folder )
6.     Go to step 2 and retrieve the files from this folder
7. end foreach

```

Equation 2: Crawling algorithm pseudo code

```

private void listFolder(File file)
{
    //Get all the folders and (.doc, .ppt, .xls and .pdf)files from the file
    File[] files = file.listFiles(new CrawlerFileNameFilter(crawlFileTypes));

    for (int i = 0, n = files.length; i < n; i++)
    {
        //File is a directory so crawl this as well
        if (files[i].isDirectory())
        {
            listFolder(files[i]);
        }
        else if (!crawledFiles.contains(files[i].toString()))
        {
            //Check if this file has already been crawled
            //all_crawled_files contains all the files
            //crawled by the system
            CrawledFile exists =
                all_crawled_files.exists(files[i].toString());

            if(exists == null)
            {
                //This file has not yet been crawled,
                //so save all necessary data
            }
            else
            {
                //This file has been crawled already
                //Check if file has changed
                if(exists.getLastModified() != files[i].lastModified())
                    //Update the crawl-file info and
                    //place file on TO-BE-Parsed cue
                else
                    //The file has not changed so do nothing
            }
        }
    }
}

```

Code Box 1: Outline of the Crawler's recursive "listFolder()" method

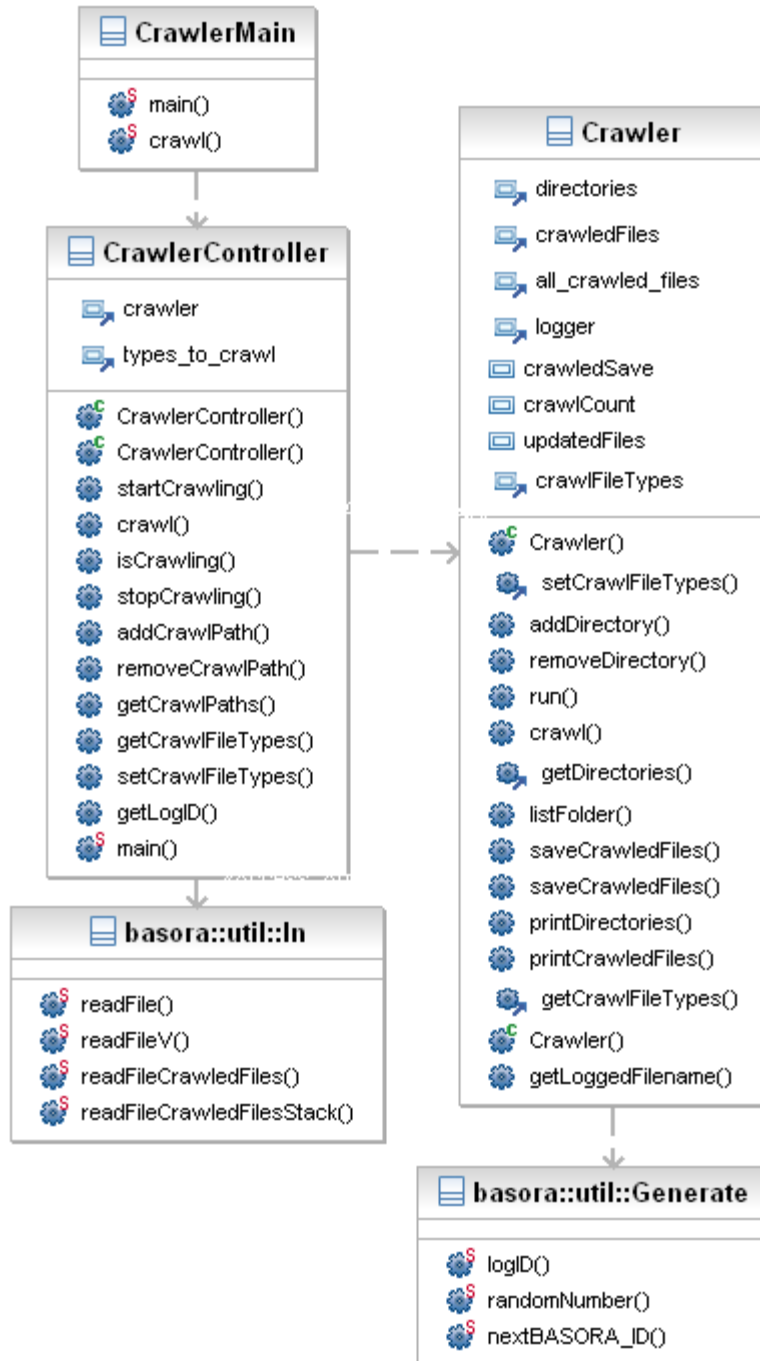


Figure 21: Class Diagram Crawler (The arrows indicate dependency)

Parsing

The parser is capable of parsing the file types: Microsoft Word, Microsoft PowerPoint, Microsoft Excel and PDF. The Microsoft Office file types that are supported are only the Office 2003 and lower versions. Office 2007 file type support was not implemented in the current version of Basora Enterprise Search, this was left out because it was not a vital requirement according to the stakeholders as discussed in paragraph 5.2.1. The analysis of the dataset was done during the implementation phase in order to be able to make decisions about file type support and such.

The parser's main purpose is to extract the metadata and contents from a document; the metadata usually contains the date created, date last modified, file type, author, title, keywords, etc. The first three fields are filled in automatically by the system so these are always available. The last three are not always accurate or even available. The author field is usually filled in by the system, but is not always correct. It is often empty or contains the login name of the computer on which the document was created. So extracting this, another way could help ensure the accuracy of this data. This was outside the scope of this research and was not pursued.

The figures at the end of this paragraph show the class diagrams of the parsers objects and classes. In order to start the parsing procedure the following steps need to be taken.

- Instantiate the "ParserController" class
 - Its constructor automatically retrieves the "CrawledFile" objects that need to be parsed
- Run the parser
 - Just execute the "process()" method.

The parser also has the option to be run in its own thread this is done by using the "process(true)" method. As stated in paragraph 5.2 the parsing process is split up into a Cacher and a Parser itself. The process of parsing the documents goes as follows: first 100 documents are stored to the local drive, these are then parsed by the system and the next 100 documents are done in the same fashion (The choice for 100 documents was discussed in paragraph 5.2.1). The "Parser" class method "process()" takes care of this part of the process. While the "parse()" method actually parses the documents, this means extracting the metadata and text from the document and placing it into a "RichDocumentData" object. Once this is done it is converted to XML formatted for indexing by the Solr indexer, this is done by the method "getIndexableXML()". This is then saved to the local disk so the committer can send it to the Solr Indexer. The basic structure of the "RichDocumentParser" interface and the classes "PowerPointParser", "WordParser" & "ExcelParser" are based on a Rich Document Parsers implementation obtained from apache POI open source subproject posted by Eric Pugh.

The "parseDocument()" method seen in the Parser class, is responsible for sending the documents that need to be parsed to their designated parser, either to the "PowerPointParser", "WordParser", "ExcelParser" or the "PDFParser" each of these implement the "RichDocumentParser" interface. Equation 3 shows the pseudo code of the algorithm used for the parsing process. The system has to parse all the files that were crawled, so it does this by repeatedly copying 100 of the crawled files to the local file system, then parsing these, deleting them. This process is repeated until all files have been parsed.

```

1. while(crawled files available)
2.   copy 100 of these files to the local disk
3.   foreach ( copied files )
4.     Check if its file type and run the corresponding parser
5.   end foreach
6. end while

```

Equation 3: Parsing Process - Pseudo Code

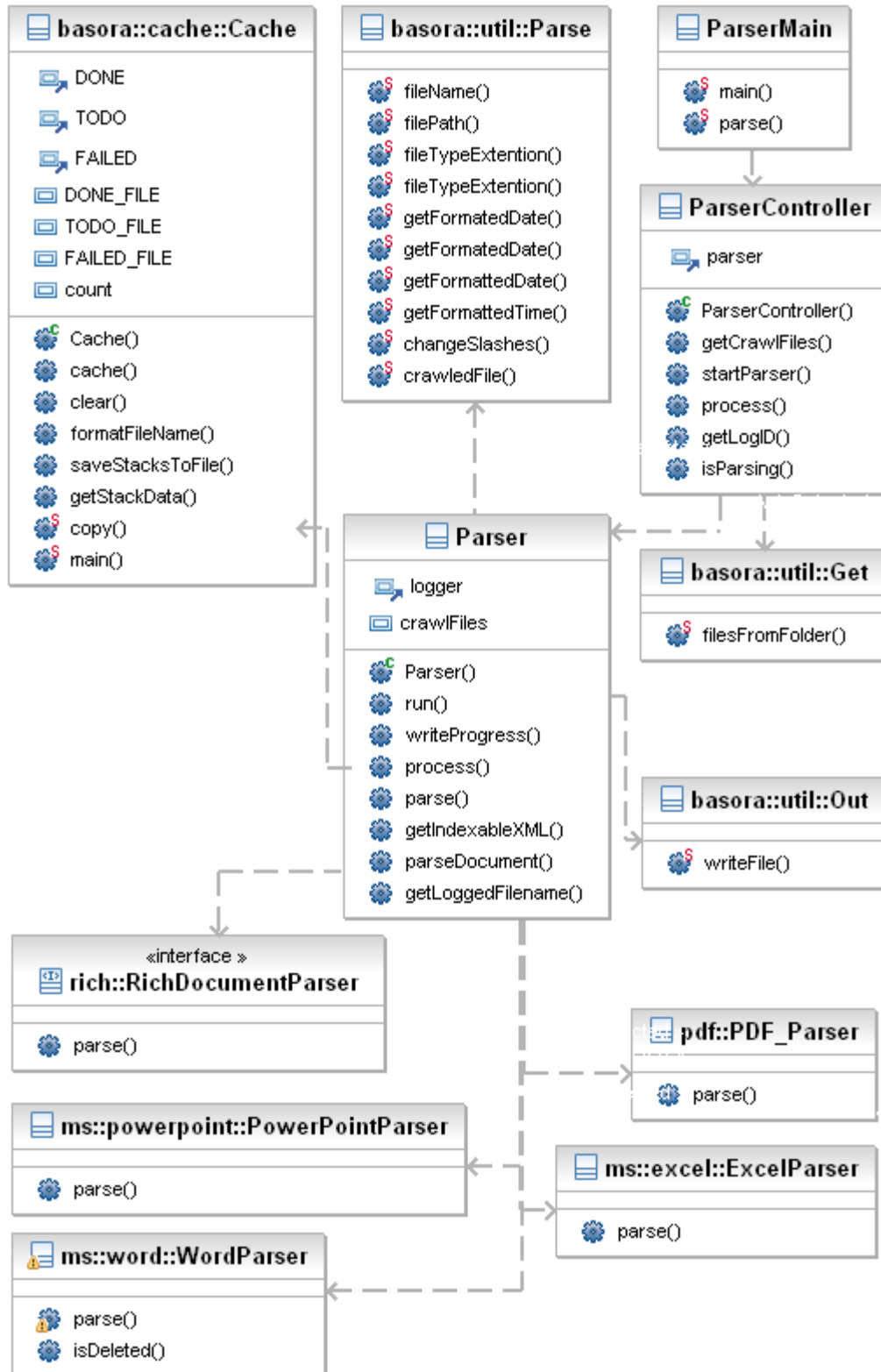


Figure 22: Parser Class Diagram (The arrows indicate dependency)

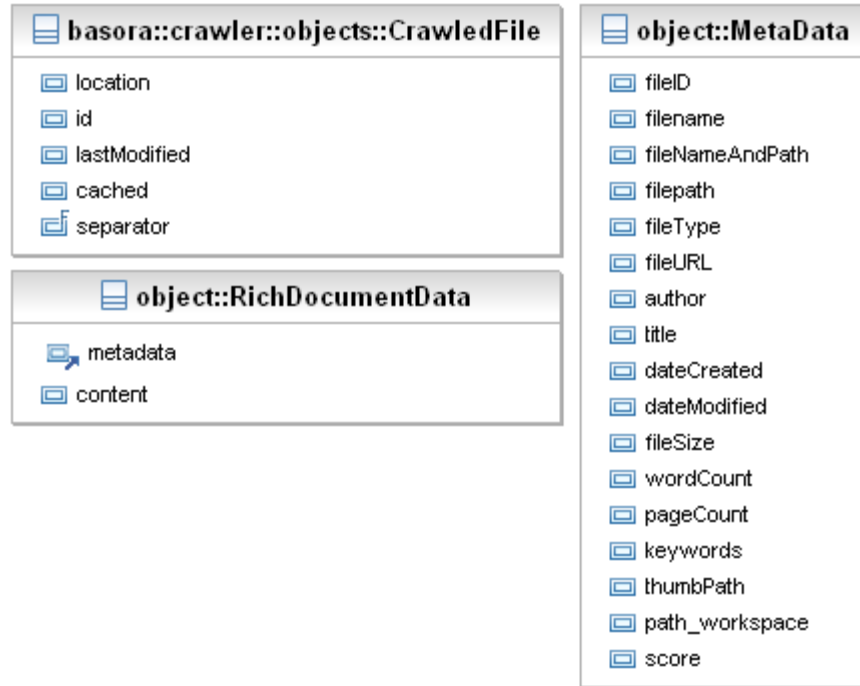


Figure 23: Objects used by the parser (Getters and setters were left out)

Committing

The committer is the bridge between the crawling & parsing process, and the Solr Indexing process. Its job is to send all the generated indexable XML files to the Solr Indexer, this is done by taking the following steps:

- Instantiate the “CommitController” class
 - Its constructor automatically Instantiates the “Commit” class
- Run the committer
 - Just execute the “commit()” method.

As for the crawler and the parser the committer can also be run in its own thread, this is done by executing the method “commit(true)”. At the end of the commit process the “optimize()” method is called which allows the Solr indexer to optimize the index. All the methods and classes referred to are listed in the class diagram that is available in Figure 24. The inspiration for the basic structure of the committer comes from a class that is available with the Apache Solr Java library, namely the “org.apache.solr.util.SimplePostTool.java”. Equation 4 shows the pseudo code of the committing process. Each of the indexable xml files that were produced by the parser are posted to the Solr Indexer.

```

1. foreach (parsed - indexable xml file)
2.   Post the xml file to the Solr Indexer
3. end foreach
  
```

Equation 4: Committing Process - Pseudo Code

Pre-processing

The pre-processing steps discussed in the previous paragraphs need to be executed beforehand, so the search engine can function. The complete process is described in Equation 5, first the administrator has to choose a folder from a file system that needs to be pre-processed for searching. Then the crawler is run over the selected file system, retrieving all the Word, PowerPoint, Excel and PDF documents. Next the parser extracts the necessary metadata and the contents from the files and formats this into indexable XML. Finally the parser commits all these files to the Solr Indexer, so it can build the search engine's index.

1. Select folder from file system to pre-process
2. Run Crawler on selected folder
3. Run Parser on the files that were crawled
4. Run Committer for all indexable XML files

Equation 5: Pre-Processing - Pseudo Code

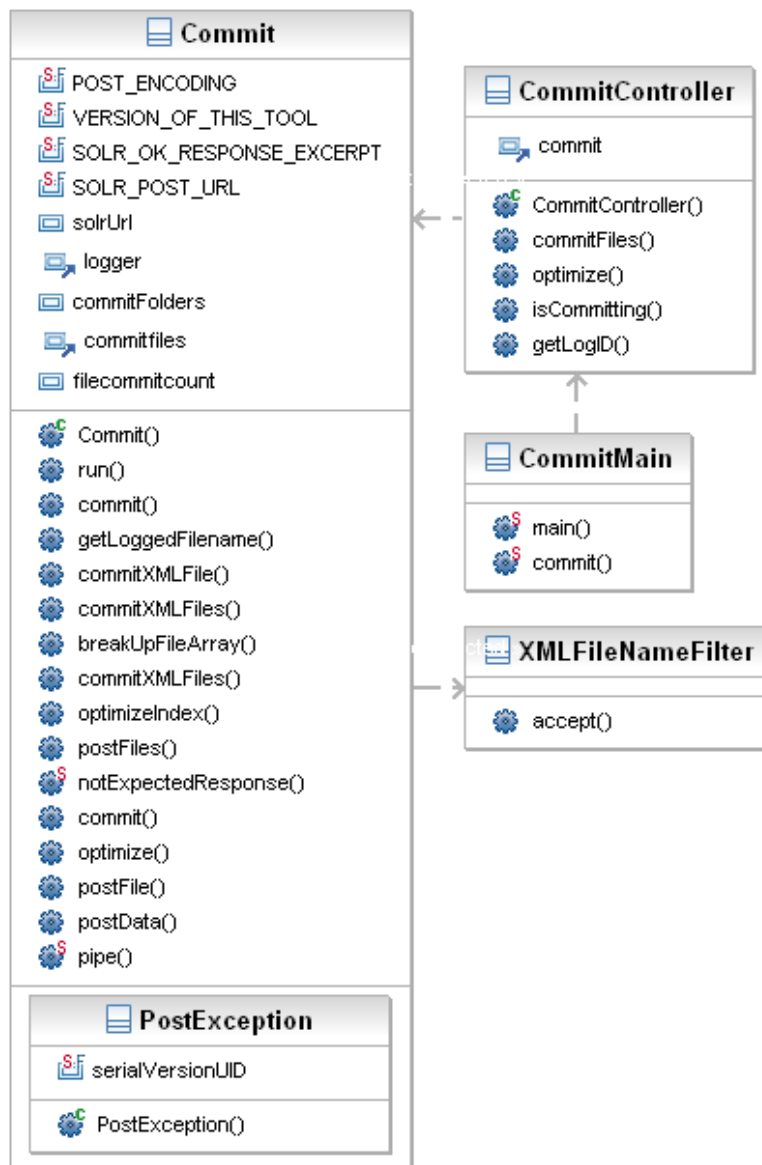


Figure 24: Committer Class Diagram (The arrows indicate dependency)

6.2.2 Solr Configuration

In order to make the Solr Server function as the Basora Enterprise Search prototype's, Indexer and Searcher, several configuration steps were needed. First the Solr server needed to be unpacked and configured for it to run on the Apache Tomcat 5.5 server. This was done by creating a folder on the local disk and unpacking the configuration files the Solr server needs into this. Then within the Apache Tomcat 5.5 Interface a reference was added as to the location of this folder. Hereafter the configurations files "solrconfig.xml" and "schema.xml" were customized to the needs of the BES prototype. The main customization done in the "schema.xml", this entailed configuring what data the server was going to be receiving and in what format. The format used can be seen in Code Box 2, here the fields that will be indexed are all defined.

```
<field name="id" type="string" indexed="true" stored="true" required="true"/>
<field name="type" type="text_ws" indexed="true" stored="true"/>
<field name="name" type="text" indexed="true" stored="true"/>
<field name="path" type="string" indexed="true" stored="true"/>
<field name="title" type="text" indexed="true" stored="true"/>
<field name="author" type="text" indexed="true" stored="true"/>
<field name="created" type="text" indexed="true" stored="true"/>
<field name="modified" type="text" indexed="true" stored="true"/>
<field name="filesize" type="text" indexed="true" stored="true"/>
<field name="keywords" type="text" indexed="true" stored="true"/>
<field name="data" type="text" indexed="true" stored="true"/>
<field name="pagecount" type="string" indexed="true" stored="true"/>
<field name="wordcount" type="string" indexed="true" stored="true"/>
<field name="thumb" type="string" indexed="false" stored="true"/>
```

Code Box 2: Format of the index-able XML files to be received by the Committer

The "id" is a field is used to store a unique identifier of the document, while the field "type", is used to store the file type. Furthermore the fields "name" and "path", respectively store the file name and the location of the file (its path). Now the fields "title" and "author" store the title of the document along with the name of the author. The date a document is created and last modified is stored in the fields "created" and "modified", and its file size is placed in the field "filesize". Now the keywords that have been added to a document as metadata are stored in the "keywords" filed. In the field "data" all the text from a document is stored, this is needed in for the Basora Engine (discussed in paragraph 5.2.3) to be able to calculate the textual document summary Top Ranking Sentences. A documents page and word count can be stored in the fields "pagecount" and "wordcount". The last field shown in the code box is "thumb", this is used for saving the location and name of the thumbnail that has been generated and stored.

For all these fields it is also possible to indicate whether the server should index them or store them. Setting the "store" variable to true, informs the Searcher to retrieve this field when returning the search results. Only the "thumb" field is not being indexed because this file location and name does not contain any valuable information about the document. When setting the indexed field to true, not only does the Searcher index it but it also allows a user to specifically search through these fields (e.g. by sending a query in the form "fieldname: query-terms").

After configuring the Solr Server settings, finally a preconfigured Solr Server WAR file was deployed on the Apache Tomcat 5.5 Server, making the indexer and searcher ready for use.

6.2.3 Basora Engine

While the Solr Indexer and Searcher provide the necessary Indexing and Searcher components for the Basora Enterprise Search prototype. BES also contains an engine that ensures proper communication between the two. This is done mainly through the “EngineController” class, its class diagram is available in Figure 25. The main result retrieval methods within this class are “getResults()”, “getNextResults()”, “getFilteredResults()” and “getSortedResults()”. Their main purpose is retrieving the results from the Solr Searcher, extracting the snippet and TRS (TRS extraction is discussed in paragraph 6.2.6), if necessary sort or filter the results, and finally converting these to HTML, to be able to display them in the user interface.

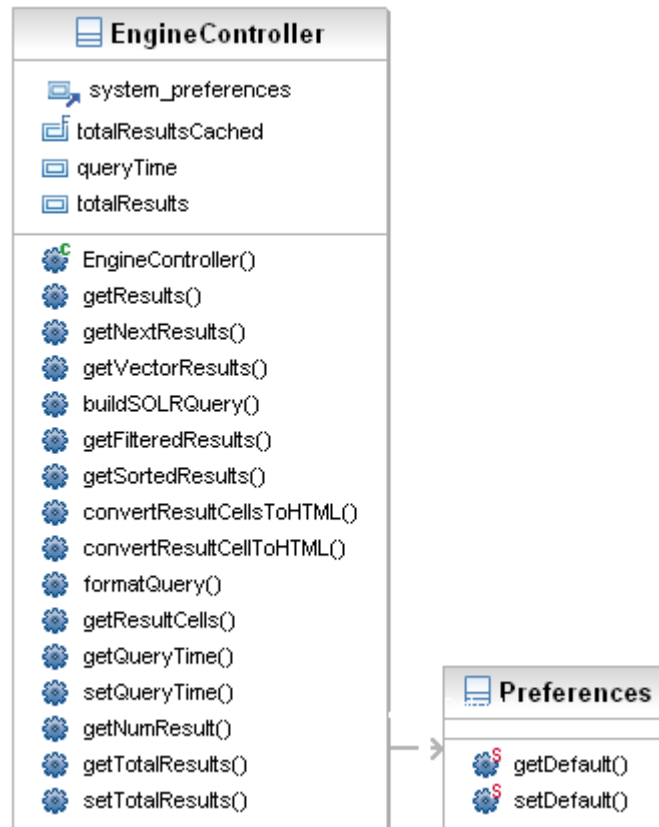


Figure 25: Basora Engine Controller Class Diagram

The following methods each share some responsibility in performing this process, namely the “getVectorResults(query, start, end)”, “buildSOLRQuery()” and “convertResultCellsToHTML()”. The first method “getVectorResults(query, start, end)” sends the user’s query to the Solr Searcher and retrieves the XML results. Because the Solr Searcher always retrieves the first result to the one specified, this method has to, extract the results specified by the start and end parameters from the retrieved XML. Then these are processed into “ResultCell” objects (Figure 26), using a custom XML parser developed for a previous project. This XML parser performs about 10 times faster than the standard Java XML parser, because it only contains the basic components necessary and it has been efficiently implemented (as discussed in paragraph 5.2.3). Finally the “getVectorResults(query, start, end)” method returns a Vector containing the “ResultCell” objects corresponding to the query that was submitted. The method “buildSOLRQuery()” formats the query submitted by the user, by adding the

necessary parameters before it is sent to the Solr Searcher. And the last method namely “convertResultCellsToHTML()” is used to convert the retrieved “ResultCell” objects into formatted HTML.



Figure 26: “ResultCell” Class

6.2.4 Web Interface

This paragraph discusses the implementation of the Basora Enterprise Search (BES) prototype’s Web Interface. First the interface intended for the users of the BES prototype is elucidated, followed by an explanation of the interface for BES Administrators.

Interface for Users

The Basora Enterprise Search prototype user interface was developed using HTML 4.01 and CSS 2.0. During the design phase the decision was made to keep the interface as clutter-free as possible. It was based on the IBM Omnifind interface shown in Figure 8. The starting page of the user interface of the BES prototype is shown in Figure 27.

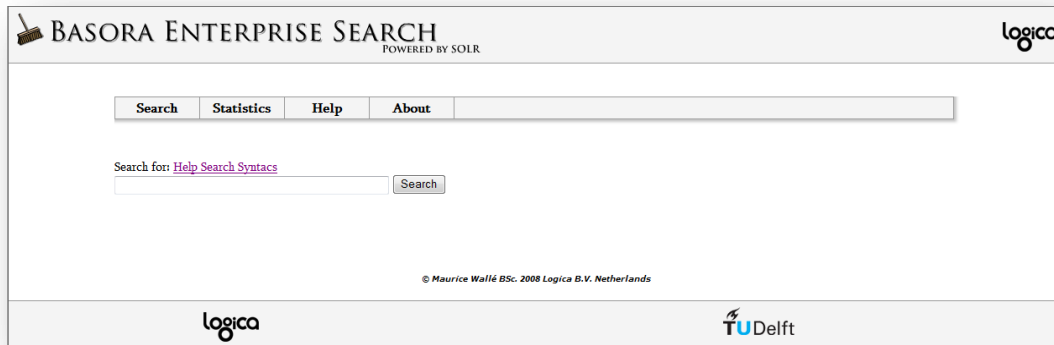


Figure 27: Basora Enterprise Search prototype main interface

In Figure 28 and Figure 29 the Statistics and Help interface are presented. The statistics interface shows the user several variables that are registered by the Basora Search Engine. It registers how many times the sort and filter functions are used, how many queries have been submitted and the amount of result pages that have been viewed.

The help interface focuses on the main differences between regular Enterprise Search Engines and Basora Enterprise Search, namely the additions TRS and Thumbnail. Their main purpose is explained and a few simple examples are given of the search engine syntax.

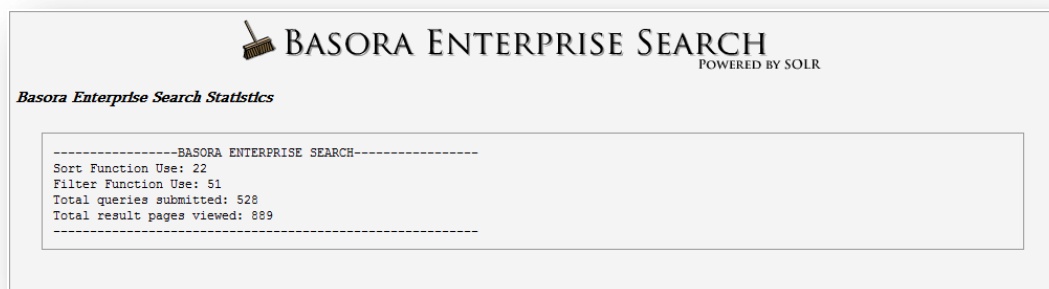



Figure 28: Statistics interface



BASORA ENTERPRISE SEARCH

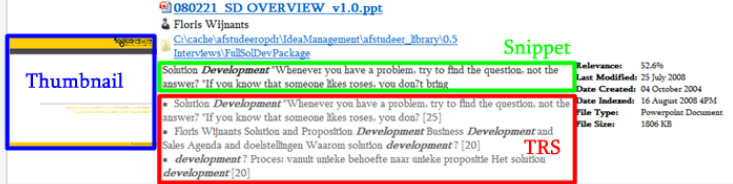
POWERED BY SOLR

Basora Enterprise Help

Basora Enterprise Search offers you all the searching capabilities of a standard full text search engine.

Some Examples:
 "one two" - will tell the engine to return all documents containing the words "one" and "two" adjacent to one another
 one two - will return all documents containing either the word "one" or "two" or both.

The following section provides help on the layout that is used for presenting the search results.



Snippet
 A snippet is the first piece of text from the document that contains your query terms.

Thumbnail
 A thumbnail is a small image of the first page of the document. In this version of Basora Enterprise Search thumbnails are only shown for PDF and PowerPoint documents.

TRS
 TRS stands for Top Ranking Sentences, these are up to three sentences found in the document that contain your query terms. They are ranked according to where they are located in the document, how many of your query terms it contains, and how many times your query terms appear in the sentence.

These sentences give you a better overview of a document, so you do not have to open this to know what it contains.

Figure 29: Help interface



BASORA ENTERPRISE SEARCH

POWERED BY SOLR

Status: **Prototype**

Version: **0.3**

Developed by: © **Maurice Wallé BSc.**



Figure 30: About interface

Figure 30 shows the about interface, it only contains a version indication along with the status of the software, and the name of the developer. In Figure 31 the sorting and filtering functions interface is presented. This is always available to the user when looking through search results, so the user can always filter the results by document type or sort them. The results can be sorted by: "Relevance", "File type", "Title", "Author", "File size", "Date created", and "Date modified", additionally the sorting order can also be specified. The sorting options available within the Omnifind system are "relevance", "date", "entry date", "entry size", "file size" and "modified date"; some of these seemed confusing to the

stakeholders. The sorting options were partially based on the ones available within the Omnifind environment, but the stakeholders also made several suggestions. They found that it would be essential to have the sort options available directly with the search results.

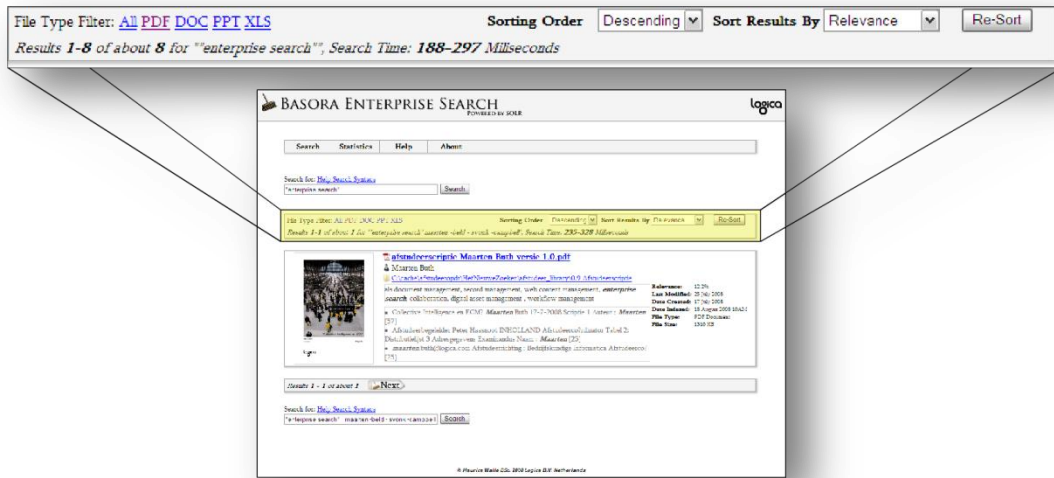


Figure 31: Sorting and Filtering functions interface

Administrator Interface

This paragraph gives an overview of the features available in the administrator interface. In order to get to this interface the administrator has to go to the “admin.jsp” page manually. The main differences between the administrator interface and the client interface is the additional menu items: “Preferences”, “Evaluation”, and “Settings” this is shown in Figure 32.

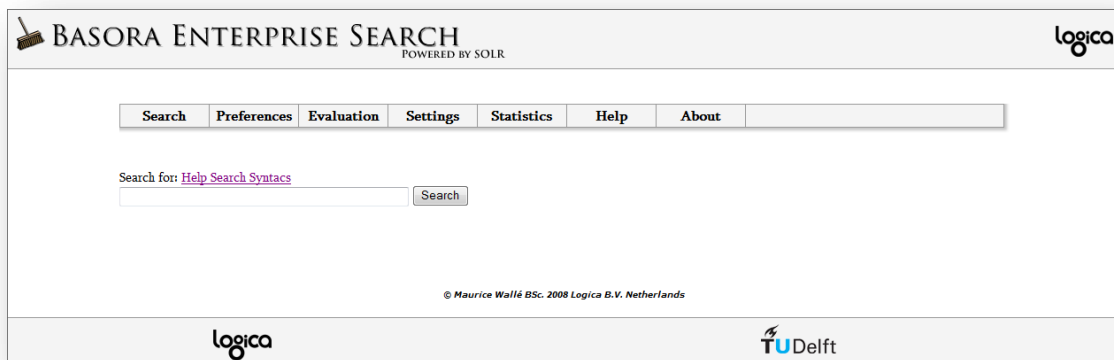


Figure 32: Administrator Interface

Search Result View:	4. Baseline-TRS-Thumbnail ▾	Number of results per page:	10 ▾
Sorting order:	Descending ▾	Sort Search Results By:	Relevance ▾
Save		Cancel	

Figure 33: Preferences Interface

Figure 33 gives an overview of the preferences interface; here the administrator can set all the default options for the preferences: Search Result View, Sorting order, Sort Results By, and Number of search results per page. The following paragraph shows the implemented search result presentation layouts available within the Basora Enterprise Search prototype interface.

6.2.5 Search Result Presentation Layouts

The results can be displayed in four layouts different layouts a choice between the four can be made within the preferences page (Figure 33). The choices for the different layouts 1 to 4 respectively are Baseline, Baseline & TRS (Top Ranking Sentences), Baseline & Thumbnail and Baseline, TRS & Thumbnail. An overview of each layout is visible in the figures listed below.

Figure 34: Layout 1: Baseline

Figure 35: Layout 2: Baseline & TRS

Figure 36: Layout 3: Baseline & Thumbnail



Figure 37: Layout 4: Baseline, TRS & Thumbnail

The first layout is a baseline and only contains the standard information that most search engines provide with their results, this information is available in all four layouts. A snippet is basically the first piece of text from the document that contains the query terms used. Layout 2 consists of the baseline and added to this are the top ranking sentences (TRS), these are maximally 3 sentences that have been ranked according to three characteristics as explained in paragraph 5.2.5. Layout 3 consists of only the baseline and a thumbnail of the first page of the document, this thumbnail can help the user identify the document that is being looked for (as explained in paragraph 3.1.2). Layout 4 is the combination of both the textual and visual layout elements namely the baseline, the Top Ranking Sentences and the Thumbnail. This layout gives the user the most information about the returned documents.

6.2.6 Basora TRS Calculation

This paragraph discusses the TRS calculation process that is done by the Basora Engine within the “TRS” class (the TRS calculation algorithm is presented in paragraph 5.2.5); additionally the snippet extraction that is done in the “Snippet” class is also explained. The class diagrams of these classes are shown in the figure below.

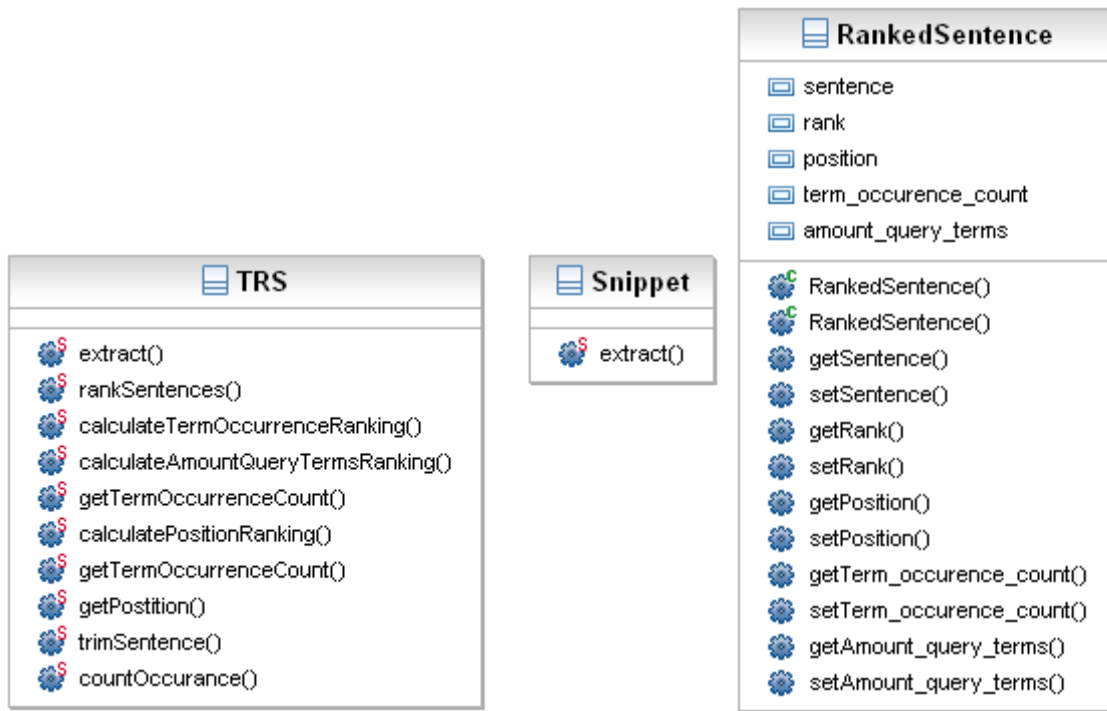


Figure 38: TRS and Snippet Classes & Ranked Sentence Object

Once the XML results are returned by the Solr Searcher these are processed into “ResultCell” objects, the Searcher also returns up to 9 text fragments that contain the query terms. The first one returned is the first occurrence of the query terms so this fragment is sent to the “extract()” method in the “Snippet” class this converts this to HTML so it can be presented to the user.

To calculate the Top ranking Sentences the text fragments get sent to the TRS “extract()” method hereafter the method returns the TRS converted in display ready HTML. The calculation process is discussed in paragraph 5.2.5, the calculation phases and the methods that carry out the calculations within the “TRS” class will now be discussed.

The “calculateTermOccurrenceRanking()” method calculates a ranking based on the amount of times the query terms occur within each of the sentences and divides this by the total amount of occurrences over all the sentences this is then multiplied by 100. A very simple example is given in the example box below. As shown here the third sentence has the highest term occurrence ranking value.

```
Initial data:
Query:      search using basora

Sentence 1: basora search is a very valuable tool
Sentence 2: using search in every day applications
Sentence 3: basora enterprise search prototype has been developed using
Sentence 4: using the latest technology we plan to develop

Total amount of occurrences: 8

Rankings
Term Occurrence Ranking Sentence 1: (2/8) * 100 = 25
Term Occurrence Ranking Sentence 2: (2/8) * 100 = 25
Term Occurrence Ranking Sentence 3: (3/8) * 100 = 38
Term Occurrence Ranking Sentence 4: (1/8) * 100 = 13
```

Example Box 1: Term Occurrence ranking example

The “calculateAmountQueryTermsRanking()” calculates a ranking based on how many of the query terms occur in each of the sentences and divides this by the total amount of query terms submitted by the user and this is then multiplied by 100. In Example Box 2 a simple example is given of how this ranking is calculated. Here again sentence 3 has the highest ranking.

```
Initial data:
Query:      search using basora

Sentence 1: basora search is a very valuable tool
Sentence 2: using search in every day applications
Sentence 3: basora enterprise search prototype has been developed using
Sentence 4: using the latest technology we plan to develop

Total amount of query terms: 3

Rankings
Amount Query Ranking Sentence 1: (2/3) * 100 = 66
Amount Query Ranking Sentence 2: (2/3) * 100 = 66
Amount Query Ranking Sentence 3: (3/3) * 100 = 100
Amount Query Ranking Sentence 4: (1/3) * 100 = 33
```

Example Box 2: Amount query terms ranking example

The “calculatePositionRanking()” calculates where the a ranking based on the relative position of the sentences in the document. This is done by ranking the sentences based on the following criteria: If the sentence occurs in the first 20% of the document it gets 50 points, if it occurs in the last 20% of a document it gets 40 points, and otherwise it gets 10 points. The reasoning behind this calculation is explained in paragraph 5.2.5.

The calculation methods discussed above are all used in the “calculateRanking()” method, which ensures all these calculations are done and that one total ranking score is calculated for each of the sentences. This method again is used by the “rankSentences()” method which has to return the top three top ranked sentences formatted in HTML with their rankings appended to the end. It first sorts the `java.util.Vector` containing the ranked “RankedSentence” objects and returns the top three.

6.2.7 Basora Searching

The Basora search engine supports basic full text search, so users can submit queries to the engine just as they would in Google. A few examples being: When searching for all documents containing the word professional and intern the query that can be provided is [Professional Intern]. Additionally when searching for all documents where these words are adjacent to one another the query terms need to be placed between quotes e.g. [“Professional Intern”].

6.2.8 Future Additions

This paragraph gives a list of future additions that could still be added to the Basora Enterprise Search (BES) prototype. Adding these components would extend BES prototype’s file type support and several other features that were not essential within this research project. The BES prototype code is documented and well structured; additionally there is Javadoc available that covers the essential components of the system. This is useful because the BES platform can be used by others to continue contributing to the evolving science of Enterprise Information Retrieval.

These are several future additions that can still be added to the Basora Enterprise Search Prototype:

- Extract thumbnail from DOC, DOCX, XLS, XLSX, PPTX files.
 - Can be resolved either finding a java library for these purposes or converting these files to PDF and per curing the thumbnail using the PDF thumbnail extractor method.
- Extract data from XLSX, DOCX, PPTX files
 - Can be fixed by converting these to PDF
- Log all Queries submitted so these can be used as auto-complete data
- Automatic extraction of metadata from documents (Title, Author, Keywords, etc.)
- Multilingual character support
- Adjust the TRS ranking algorithm
 - By researching what additional variables could be used
 - And how the current ranking algorithm can be optimized
- Multilanguage analyser for indexing and searching.
 - This would allow for better indexing of the data.

7 Evaluation

In this chapter the Evaluation Phase of this project is discussed. First the specifications of the computer that was used during the evaluation is presented; then an overview is given of the fixed dataset that is used. Furthermore the performance evaluation procedure is discussed, and finally the user evaluation procedure using the IMPACT model for user evaluation is elucidated.

7.1 OS, Hardware & Software

This paragraph gives an overview of the hardware and software that was used during the testing phase. The computer used is a Lenovo Thinkpad T-series: Intel Core 2 Duo T7500 2.20 GHz (2 CPU's), 800 MHz FSB, 2 GB of RAM, 100 GB Hard disk drive, and runs Windows XP Professional Edition SP2. This system was used for both the performance and user tests in order to keep the hardware and software variable constant throughout the evaluations. Furthermore Apache Tomcat 5.5, Solr, and Basora Enterprise Search were installed on this computer. All technical details of the machine used are listed in Table 4.

Table 4: Full Hardware and OS specifications of the evaluation computer

Description	ThinkPad T61
Operating System	Windows XP Professional Edition with Service Pack 2
Bus type	PCI/PCI Express
Screen size	15.4 inches
Maximum resolution	1680x1050
Maximum colours	16777216 colours
Graphics chipset	Intel Graphics Media Accelerator X3100
Processor	Intel® Core 2 Duo T7500
Processor Speed	2.2 GHz (2 CPU's)
Front Side Bus	800 MHz
Memory (RAM)	2 GB
Hard disk type	Serial ATA
Hard disk size	100 GB
Network card	Integrated Intel PRO/1000 Gigabit
Keyboard type	Full size Keyboard
Mouse	Standard Lenovo Optical Mouse
Dimensions	29.7x357.5x255 mm
Weight	2.67 Kg

The server and client side of the Basora Enterprise Search engine were both run on this machine to ensure that network lag did not influence the test results, especially of the performance test.

7.2 Dataset

The dataset used comes from an Enterprise Content Management system dedicated to interns of Logica (as discussed in paragraph 1.3) where they can store all sorts of data. Of which the most essential data consists of the interns their thesis work, plan of approach, factsheet and such. This dataset was readily

available and the prototype is now in use making this dataset searchable for the interns and employees at Logica.

During development phase a tool was created to cache all the data from dataset mentioned earlier to a local disk. After doing this a breakup of the contents of the data was made. In Table 5 an overview of the size of dataset and the file types it contains is presented. It also shows the total parsing errors. These errors occur because either the files are corrupted or the parser cannot handle it because the file is too large for the Apache POI or the PDFBox library to handle both for parsing and thumbnail generation. These open source libraries are both unfinished releases, seeing that they performed acceptably and were the only libraries with the necessary tools. They were chosen to be used for the development of the Basora Parser.

Table 5: Document type totals of the evaluation dataset

Document Type	Crawled	Parsed	Committed
Word Documents	2856 (47%)	2706 (45%)	2706 (50%)
PowerPoint Documents	343 (6%)	270 (5%)	270 (5%)
Excel Documents	692 (11%)	442 (7%)	442 (8%)
PDF Documents	2034 (34%)	1990 (34%)	1990 (37%)
Office 2007 Documents	140 (2%)	-	-
Total Errors	-	517 (9%)	0
Total Documents	6065	5925	5408

The percentages listed between the brackets are based on the total documents listed in the same column.

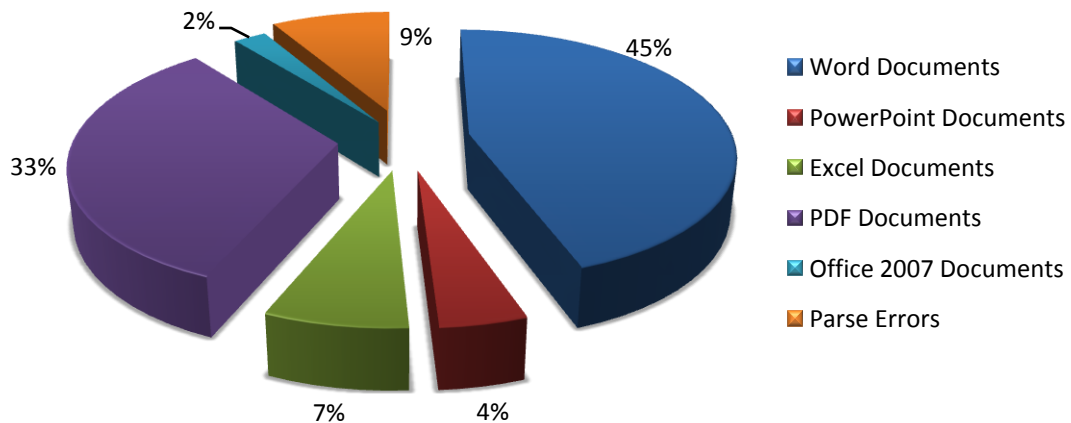


Figure 39: Pie Chart of the evaluation dataset

The dataset consists of about 6065 documents, of which 2% are Office 2007 documents. This leaves about 5925 Microsoft Office 2003 and PDF documents, of which about 9% cannot be parsed by the system. The total amount of documents that the system can parse and index is 5408, this is about 90% of the documents available on the working tomorrow workspace. This was deemed an acceptable ratio to be able to complete the prototype evaluation, during the final prototype assessment meeting discussed in paragraph 5.1.

7.3 Performance Evaluation

The performance evaluation assesses some basic aspects of the prototype, namely the time it takes to pre-process the dataset, response time of the search engine and the average number of results it returns. These aspects were taken from an Enterprise Search Engine evaluation guide by Google [P11]. This document describes seven critical characteristics of an Enterprise Search Engine. Only three aspects were evaluated, because these were needed to indicate how the Basora Enterprise Search (BES) prototype performs compared to current Enterprise Search Solutions. Some of the other characteristics mentioned in the evaluation guide were: Security, How often it crawls the content, Costs and maintenance fees, these were outside of the scope of this evaluation.

In order to establish a baseline IBM Omnifind 8.5 Enterprise Search engine was put through the same evaluation steps as the BES prototype. First both systems were installed on a machine with the specifications listed in paragraph 7.1. Then both systems were put through their pre-processing steps and clocked. Hereafter the last two aspects response time and average number of results were evaluated. This was done by first gathering thirty query terms from users that actually search the dataset (the dataset mentioned in paragraph 7.2) regularly. These were used to evaluate the aspects, response time and average number of results returned. The results are available in the following paragraph along with a discussion as to what these results indicate.

7.4 User Evaluation

In this paragraph the set up of the user evaluation is explained, the model that is used is presented, each of the steps presented in the model are worked out. Finally the evaluation plan is presented; here the actual user evaluation procedure is described.

7.4.1 IMPACT

IMPACT is a model for user evaluation described in Benyon et al 2005 [B1]. Where I.M.P.A.C.T. is an acronym that stands for: Intention, Metrics, People, Activities, Contexts, and Technologies, this model is used to evaluate the prototype. This model was chosen because it fit the context of the user evaluation, is well documented and because of its scalability. Other methods were considered, most of these were not very well documented and only available through research articles. The results that are obtained are discussed in the following chapter (in paragraph 8.2). In the following paragraphs the basic intention of IMPACT will become clear, and its application to the user evaluation procedure of the Basora Enterprise Search Engine.

7.4.2 Intention

During this phase the aim of the evaluation is determined to establish what type of data is required. The research topic presented in paragraph 1.2 stated “*Design, implement and evaluate a system for Enterprise Information Retrieval, which employs an improved search result presentation technique*” The intention of this user evaluation is to evaluate the Basora Enterprise Search (BES) prototype.

Two of the main challenges which the BES prototype is supposed to address (as mentioned in paragraph 1.1), can be summarized as, facilitating a user during query re-formulation, and improving their relevance assessment. In order to evaluate whether the prototype actually indicates an improvement over the current commercial enterprise search engine result presentation techniques; four layouts were implemented (as presented in paragraph 6.2.5); the first contains the exact same information given to the user by the IBM Omnifind Search Solution (most Enterprise Search Solution

provide this information as their results). And the other three layouts are combinations of the baseline with the addition of the textual and visual document summaries TRS and Thumbnail (as presented in paragraph 6.2.5). By evaluating the effectiveness of each of these layouts an indication can be given whether the new presentation technique (Layout 2, 3 and 4) demonstrates improvement over the Baseline. To assess this several questions needed to be formulated namely,

- Does user interaction increase when layout elements¹⁰ are added to the search results?
- Does relevance assessment increase when layout elements are added to the search results?
- What elements of the layout are most valuable to the user?
- Which layout¹¹ is preferred by the users?

Now from this these questions the intention of the evaluation procedure has been split up into four questions that can be answered by a user test. After identifying these questions we can move on to specifying the metrics needed to tackle these questions.

7.4.3 Metrics

This phase of IMPACT focuses on “what is to be measured and how?” After careful consideration several metrics were specified based on ones available in Benyon et al 2005 [B1], these metrics and measures are all adapted from the list provided in the usability standard ISO9241 part 11. In the table below a subset of the common usability metrics that are available in [B1] are presented.

Table 6: Subset of common usability metrics shown in [B1], page 276, in table 12-1

Usability Objective	Effectiveness measures	Efficiency measure	Satisfaction measure
Overall Usability	Percentage of tasks successfully completed Percentage of users successfully completing tasks	Time to complete a task Time spent on non-productive actions	Rating scale for satisfaction Frequency of use if this is voluntary (after the system is implemented)
Meets the needs of trained or experienced users	Percentage of advanced tasks completed Percentage of relevant functions used	Time taken to complete tasks relative to minimum realistic time	Rating scale for satisfaction with advanced users
Meets the needs for walk-up and use	Percentage of tasks completed successfully at first attempt	Time taken on first attempt to complete task Time spent on help functions	Rate of voluntary use (after the system is implemented)

To tackle each of the questions posted in the previous paragraph different metrics will be needed. Each of the questions is labelled respectively: User interaction, Relevance assessment, contribution of layout features and layout preference. Inspired by the metrics used for the evaluation of the Web Information Retrieval system proposed by Joho and Jose 2006 [P2], and the ones mentioned in the book Benyon et al 2005 [B1] the following metrics were specified for each of the research questions.

¹⁰ Layout elements: Top Ranking Sentences and Thumbnails

¹¹ Layouts: Baseline, Baseline & TRS, Baseline & Thumbnail, Baseline, TRS & Thumbnail

User interaction

The interaction between the user and the interface can be measured by several variables. Using Joho and Jose 2006 [P5] evaluation metrics and the usability metrics shown in Table 6 as a guide the following variables were used to gauge the user interaction:

- Number of queries submitted to the engine per session
- Number of words used in the queries
- Number of result pages viewed during the task
- Number of documents opened/clicked on per query submitted
- Number of retrieved records opened/clicked on per result page
- Time taken to complete each task.

The first five variables are all effectiveness measures and the last one is an efficiency measure. All these variables will be measured by a plug-in developed for the Basora Enterprise Search prototype.

Relevance assessment

Relevance assessment can be defined as how easy is it to actually judge the relevance of a document based on the query formulated, and the information available in the search results. In order to measure the relevance assessment of the participants of the evaluation procedure, they needed to answer several questions namely,

1. How easy is it to find the relevant documents from the search results?
2. How easy is it to find documents that contain new information?
3. How often did the document contain what you expected to find?

Each of these questions will need to be scored by the participants with a number between 1 and 7, 1 for very easy/always and 7 for very hard/never. This variable is a satisfaction measure and is totally reliant on the participant's opinions. So it is very essential that the users score these questions honestly.

Contribution of layout elements

Some of the most essential result presentation elements that are available in the four layouts (discussed in paragraph 6.2.5) are: filename, author, location, snippet, TRS, Thumbnail, file type and file size. Contribution of each of these elements will be gauged during the evaluation. This is done by asking the participants to give a score to each of these elements as to what extent it contributed to his or her decision of viewing a document from the search results. This is done using again a scale from 1 to 7, 1 being very much and 7 being very little. This again is a satisfaction measure.

Layout preferences

In order to measure the layout preference, the participants will need to rank each of the four layouts (discussed in paragraph 6.2.5) in the order of their preference. Each layout will need to be assigned a number 1 to 4, 1 being the one they preferred most and 4 being the one they preferred least. In terms of usability measures this is also a satisfaction measure.

7.4.4 People

The user evaluation needs to be performed using several participants. It is important to specify characteristics of the intended users before acquiring these. One of the characteristics is that the evaluator should be someone that will work with the system eventually.

In this case the future users of the software are Working Tomorrow interns and employees at Logica, the employees mentioned are project managers, architects and secretaries that are assigned to the Working Tomorrow programme within Logica. Furthermore four experts within the field of enterprise search that are part of the Logica workforce will also participate for in the user tests. The recommended amount of evaluating users is between 3 and 5 [B1], but for this experiment a minimum of 20 users will be used in order to increase the significance of the findings. Another point that also needs to be addressed here is how the participants will be recruited. This is quite a trivial task for this evaluation procedure seeing that there are about 15 to 20 students working in the Rotterdam Working Tomorrow department.

7.4.5 Activities

In this paragraph some scenarios are presented and from these scenarios concrete tasks are extracted to be used for the evaluation procedure. First four scenarios are stated, followed by four tasks/assignments that the evaluation participants will need to complete. These scenarios are based on actual occurrences within Logica's Working Tomorrow programme. These were created with the help of a Working Tomorrow project manager and architect.

Scenario One

A student just started his internship at Logica and has a specific topic on which he has to base his thesis work. Now knowing that there are lots of students that are doing an internship at Logica and that there are even more that have already finished their internship, he decides to search through all the documents of these students and ex-students to figure out if any of them has done work on a similar topic. This can help him by giving him the power to easily find the information needed. So he opens up his browser and queries the Basora Enterprise Search engine for terms that are associated with the topic of his research.

Scenario Two

A working tomorrow project manager has just gathered a list of suggested assignments. Before sending these to be published to attract new interns, he has to figure out which ones are very similar to assignments already completed at working tomorrow. So he starts up his browser and goes directly to the Basora Enterprise Search Engine and queries it with terms that are associated with the various research assignments he has to check and based on the results he can decide if the assignments are unique and innovative enough to make the cut.

Scenario Three

An ex-working tomorrow student is working on an important project and he remembers that he had placed some documents on the working tomorrow workspace that are relevant for what he is currently doing, but he does not remember exactly where he put it on the workspace. So he starts up his browser and surfs to the Basora Enterprise Search page and queries this with terms that the document he is looking for likely contains, and in no time he finds the document he is looking for.

Scenario Four

A student that is doing an internship at Logica finds himself in the first step of his project, which is gathering information on the topic which he is researching, instead of going directly to Google or to an online library to do this, he chooses to use the Basora Enterprise Search Engine first. Because he knows that there have been many students before him that may have already placed the information he needs on the Working tomorrow workspace. So he opens up his browser and goes directly to the Basora Enterprise Search page, then queries the engine for the required information.

After the scenarios were created the following tasks were put together for the participants to execute while evaluating the prototype. These tasks are based on actual work related example scenarios and the simulated work task approach [P2, P6]. The classifications used namely Background Information Task, Decision Making Task, Known Item Task and Topic Distillation Task, are from the simulated work task approach described in Borlund 2000 [P6].

Assignment 1: Background Information Task

The user is given the name of a student that has done an internship at Logica and is requested to find out what the name is of his project.

Then the user is given the name of a project with the objective to attain the name of the student who worked on this.

Assignment 2: Decision Making Task

Beforehand the user is given a list of assignments and the objective here is for the user to use the search engine to determine which assignments are completely different from the ones that are available on the workspace.

Assignment 3: Known Item Task

The user gets to see a particular document and has to locate this using the search engine. It is allowed for the user to look back into the document to gather more inspiration for search queries. After the document is located the user should write down where it was located so this can be verified later on.

Assignment 4: Topic Distillation Task

The user is given an image and the objective is to find all the documents related to the topic that the image presents. The found documents should be written down so these can be analysed later on.

The intention behind the selection of these tasks was to explore the different levels of a document's textual and visual elements, which were likely to be significant to complete each of these tasks. For example, a background information task is likely to involve more textual information than visual, while a topic distillation task is likely to involve visual aspects of the documents in a greater degree than other tasks [P2]. The decision making task and the known item tasks are likely to involve both aspects in a similar degree. The assignments used during the evaluation procedure are available in Appendix F: Evaluation Assignment Form.

7.4.6 Context

For this element of IMPACT the main concern is the wider social and physical context which may affect the way the technology is used. Here the setup of the experiment with regards to the setting, the physical setup and the social norms is discussed.

The environment in which the test will take place is a usual working atmosphere that is common at any office workspace. Additionally the actual evaluation will take place in a small office on the Working Tomorrow floor, in the office a computer will be set up on which the user will have to carry out the assessment of the prototype. There will also be a person present at all times to observe the user during the evaluation; the user is also allowed to ask questions before and after every task.

7.4.7 Technologies

The technologies element of the IMPACT model is the final aspect that needs to be considered before the test plan and task specification can be worked out. Here the hardware and software used will be mentioned as well as an elaboration as to why these were chosen.

As mentioned in paragraph 7.1 the basic specifications of the computer used for the evaluation of the search engine are: Intel Core 2 Duo T7500 2.20 GHz (2 CPU's), 800 MHz FSB, 2 GB of RAM, 100 GB Hard disk drive, running Windows XP Professional Edition SP2. This is a standard issue Logica laptop for all employees. The search engine itself will be hosted locally to remove the variable network speed and to ensure that all users get the same search experience.

Several variables are measured; the tools needed to measure these have been developed alongside the prototype as a plug-in, to ensure easy and reliable testing. For every test subject an output-file is produced containing the user interaction measurements discussed in paragraph 7.4.3.

7.4.8 Evaluation Plan

The intention of this paragraph is to explain how the user evaluation will take place and in what order the tasks need to be executed. The user tests will consist of four assignments namely: Background search task, Decision making task, Known item task and a Topic distillation task (described in paragraph 7.4.5). These tasks are designed based on the simulated work task approach [P6]. The next paragraph gives an overview of the evaluation procedure itself.

Procedure

Once a user enters the evaluation room, they are first asked to read the introduction page on the evaluation sheet, afterwards they need to fill in an entry questionnaire. Then they are briefed as to how the evaluation will take place. During which they are shown the four layouts (discussed in paragraph 6.2.5) of the interface and also briefed on the nature of the evaluation, and then their questions are answered.

Hereafter the users will have to carry out four assignments (as discussed in paragraph 7.4.2) in order to successfully complete the user evaluation. Each assignment (available in Appendix F: Evaluation Assignment Form) will be done with a different layout (Layout 1 to 4). In order to reduce the bias from participants performing the same tasks with the different layouts all in the same order a Graeco-Latin-Square¹² arrangement is used. This arrangement is illustrated in Table 7 [P2].

¹² Engineering Statistics Handbook <http://www.itl.nist.gov/div898/handbook/pri/section3/pri3321.htm> - last visited: 20 October 2008

Table 7: Graeco-Latin Square Arrangement used for user evaluation (T1 = Task 1, L1 = Layout 1)

User Tests	Assignment 1	Assignment 2	Assignment 3	Assignment 4
Participant I	T1/L1	T2/L2	T3/L3	T4/L4
Participant I+1	T2/L4	T1/L3	T4/L2	T3/L1
Participant I+2	T3/L2	T4/L1	T1/L4	T2/L3
Participant I+3	T4/L3	T3/L4	T2/L1	T1/L2

Each participant completes the tasks in the order illustrated in the rows of the table, so the first participant needs to complete the first task with the first layout and the second task with the second layout and so on. And for the next participant the following row will be used, once the I+3 row is reached, the following participant will execute the tasks according to the first row (I) so this order will be cycled through until all the participants have completed the evaluation.

Each task either consists of a Background search task, Decision making task, Known Item Task or Topic Distillation Task, these are explained in paragraph 7.4.5. After completing each of the tasks the user is asked to fill in a short session questionnaire. Finally at the end of the four tasks the user is debriefed and will fill in an exit questionnaire. All the questionnaires are needed to analyse the results afterwards. These questionnaires are available in the appendix of the thesis work, along with a procedure sheet where the exact process is described step by step, and the exact assignments that were completed. All of these were also translated to Dutch because this was the native language of all the participants and this would minimize the chances of a user not understanding the essence of the questions.

8 Evaluation Results

This chapter presents and discusses the results obtained during the performance (discussed in paragraph 7.3) and the user tests (presented in paragraph 7.4) that were carried out with the Basora Enterprise Search (BES) prototype. First the performance evaluation results are presented, then the user evaluation results.

8.1 Performance Evaluation

After carrying out the performance tests, evaluating the Basora Enterprise Search (BES) prototype using layout four¹³ (as presented in paragraph 6.2.5) and the IBM Omnifind 8.5 Enterprise Search Solution, the following results were obtained. The pre-processing results are shown in Table 8, here the only results presented are the time it takes each of the Search Systems to crawl parse and index the fixed dataset (presented in paragraph 7.2) used.

Table 8: Pre-processing performance results

Search Engine	Crawling	Parsing	Indexing	Total Time
Basora	4 min	3 hour 42 min	10 min	3 hours 56 min
Basora <i>No Thumbnail generation</i>	4 min	1 hour 41 min	10 min	1 hour 55 min
Omnifind	35 min	1 hour 40 min	1 hour 30 min	3 hours 45 min

Table 9: Average Response Time and Average Number of results returned

Search Engine	Response Time (ms)	Num results returned
Basora	[Solr: 161, Basora Engine: 72] Total: 233	1558
Omnifind	158	133

In Table 9 and the next set of figures, the results obtained from measuring the average response time and the average numbers of results of the search solutions are shown. Thirty search queries were gathered from users experienced in searching the dataset and these were executed on both Enterprise Search engines. The following paragraph discusses all the results obtained.

¹³ Baseline, TRS & Thumbnail added to the search result presentation

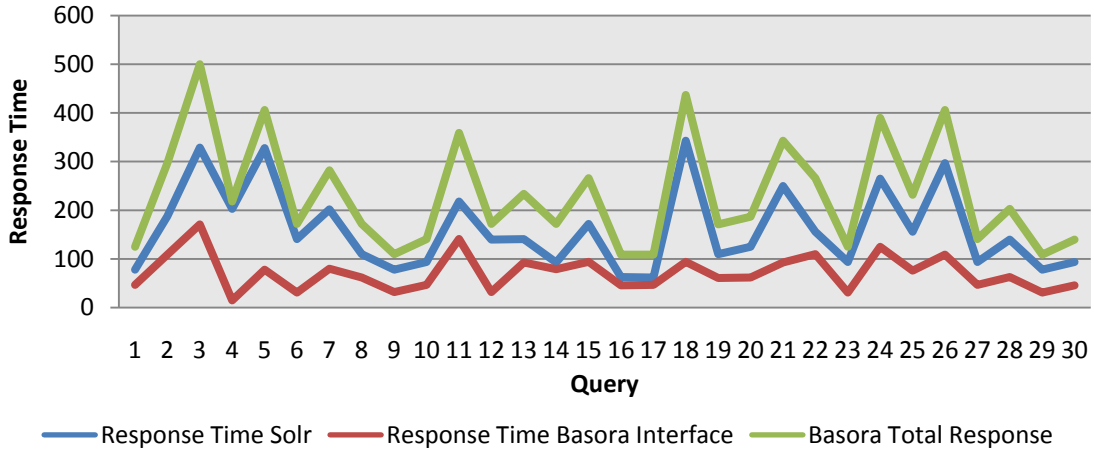


Figure 40: Basora Enterprise Search response time chart break-up

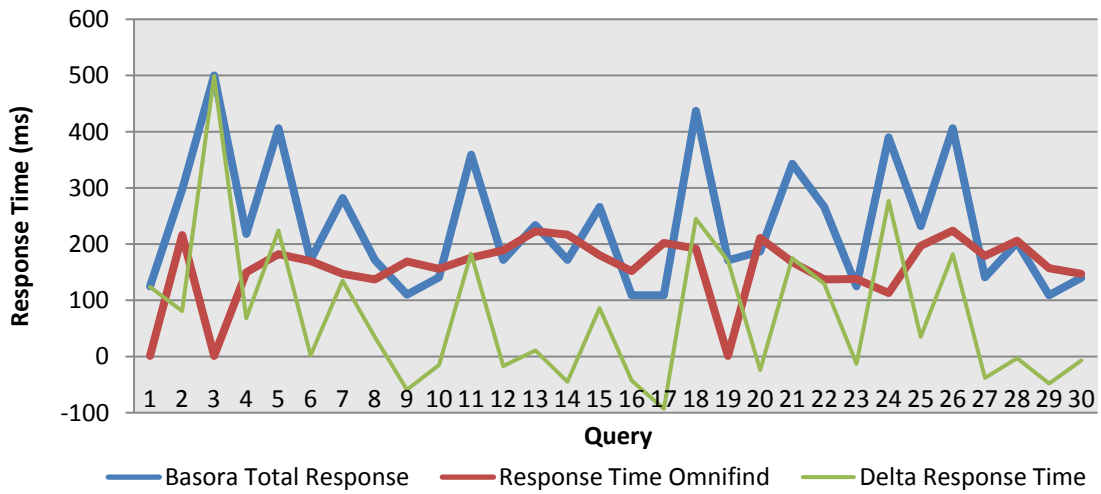


Figure 41: Basora vs. Omnifind response time

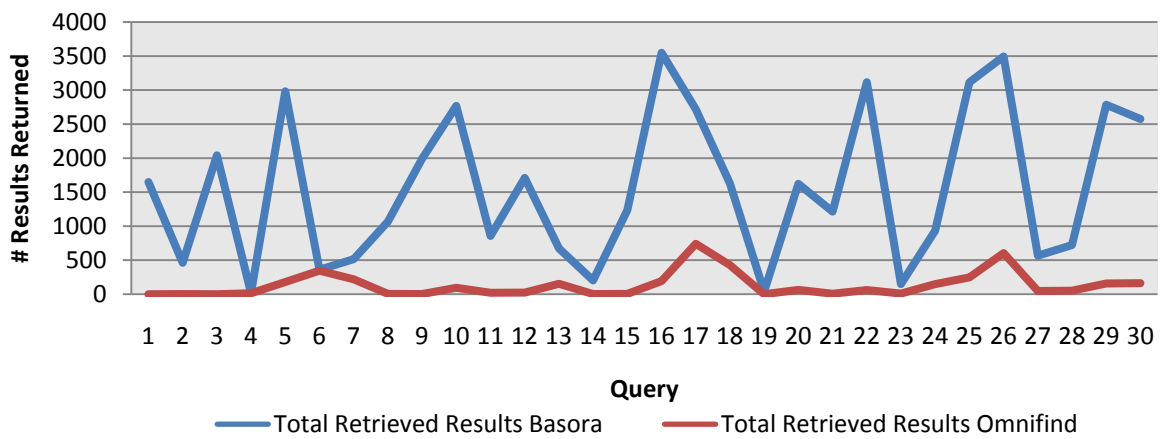


Figure 42: Basora vs. Omnifind results obtained per search query

Discussion

The results shown in Table 8 present the time it takes both search solutions to pre-process the fixed dataset (presented in paragraph 7.2). These results suggest that the Basora Enterprise Search (BES) prototype's pre-processing time more or less equal the Omnifind search solution.

There is a big difference between the BES and Omnifind Crawling and Indexing times, seeing that the exact procedure used within the Omnifind solution is not available, one could only speculate why. One of the reasons could be that BES does not contain all the security and back-up features that the commercial solutions have. One other valid reason could be that the BES solution's parsing utility directly generates indexable XML that can be posted to the indexer. This procedure is probably different within the commercial Enterprise Search solution.

For about 40% of the documents, the Basora parser needs to generate a thumbnail. This process slows down the parser significantly; this was verified by disabling thumbnail generation and running the parser once more. These results demonstrate that it only takes the parser 1 hour and 41 minutes to complete the parsing cycle. These results demonstrate that pre-processing takes about the same amount of time for the BES prototype while in the end it presents the user with more visual and textual result summaries than the IBM Omnifind 8.5 Enterprise Search Solution.

Table 9 shows the actual average numbers obtained from the performance test regarding the response time and the total retrieved documents. The response time for the Basora prototype was split into three numbers, the first one is the time it takes the Solr Searcher to retrieve the results while the second one is the time it takes the Basora Engine to process the results. Finally the last number is the total response time of the Basora prototype.

An interesting observation is that the response time for the Solr Searcher is more or less equal to the IBM Omnifind response time as can be seen in Table 9 (161 for Solr and 158 for Omnifind). This suggests that both search engines perform more or less equally in terms of response time. What also seems interesting is that the average total difference between the two is about 75 milliseconds ($233 \text{ Millis} - 158 \text{ Millis} = 75 \text{ Millis}$, Table 9) this indicates that it takes the BES prototype about 75 milliseconds more to return search results containing two extra document summary elements. In Figure 40 a break-up is shown of the response time measured for the BES prototype. This along with the figures from Table 9 indicate that an average of about 72 milliseconds of the response is accounted for by the Basora Engine. Collins 1994 [B3] states that a user considers a process to be real time if it takes less than 100 milliseconds. So this difference is not noticeable by the users, so a user will not get frustrated by the time it takes for the results to be returned.

Figure 41 shows the response time for both the Basora prototype and the IBM Omnifind engine, and the difference between the two. While the last figure (Figure 42) presents the total amount of search results obtained per search query for both the search solutions.

As presented in Figure 41 for the two queries (query 3 and 19) the Omnifind Search Engine response time dips around 0, while the Basora prototype does retrieve results for these queries. This may be because the Omnifind engine has a stricter filtering function, but it could also mean that the Omnifind engine indexes less of the documents contents than the Basora prototype does.

This evaluation indicates that the Basora prototype performs more or less equal compared to existing Enterprise Search Solutions. It also shows that the Solr Searcher performs more or less equal to an existing Enterprise Search solution and that the additions made only add an average of 75 milliseconds on top of the average response time.

8.2 User Evaluation

This paragraph focuses on the results obtained from the user evaluation. A total of thirty-two participants (6 female and 26 male) took part in the user evaluation. The entry questionnaire established that the ages varied from 21 to 50 with an average of 26.9. Their experience with web search engines varied from 5 to 12 years with an average of 9.1 and their experience with Enterprise Search varied from 0 to 10 years with an average of 2.5. From these participants four were experts in the field of Enterprise search, each with an average of 5 years of experience within this area, and an average of 10 years of experience using web search.

This paragraph presents the outcome of the user evaluation and also gives interpretations on what is indicated by the results. The measures used during the evaluation procedure are each discussed in a separate paragraph in the following order: User interaction, Relevance assessment, Contribution of layout elements and lastly Layout preference.

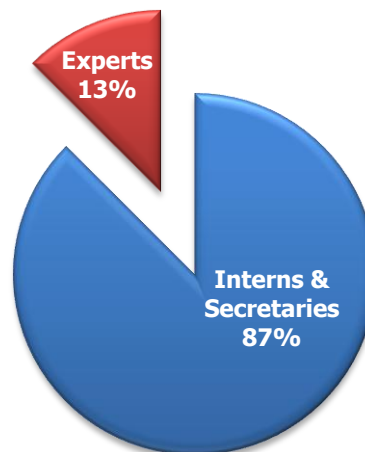


Figure 43: Participants

8.2.1 User Interaction

In this paragraph the results obtained concerning the user interaction are presented. The following result table consists of 7 columns, the first and second indicate the layout that was used and the average amount of queries that were submitted by the participants overall. Next the third and fourth columns represent the average query length and the total amount of result pages that were viewed. Furthermore the fifth and sixth columns present the amount of result pages that were viewed per search query that was submitted and the amount of documents that were clicked per result page that was viewed, and the seventh and final column shows the average amount of time it took the participants to complete a task using a certain layout. Each of these measures are averages taken over the amount of participants namely 32 ($n = 32$), except for the totals because the n here is 128. The values in each of the cells are the average and the standard deviation (between brackets).

Table 10: User Interaction

Layout	Queries submitted	Query length	Result pages viewed	Result pages per Query	Clicks per Result page	Time (min)
1	4.72 (2.32)	3.02 (0.94)	6.06 (3.60)	1.28 (0.39)	0.70 (0.47)	5.48 (2.41)
2	5.13 (2.69)	2.82 (0.81)	6.81 (4.21)	1.34 (0.48)	0.58 (0.41)	5.53 (2.16)
3	4.78 (2.43)	3.03 (1.06)	5.81 (3.69)	1.20 (0.29)	0.35 (0.32)	4.97 (2.27)
4	4.25 (2.82)	3.14 (1.85)	5.38 (3.87)	1.51 (1.77)	0.57 (0.60)	5.29 (2.75)
Total	4.72 (2.56)	3.00 (1.22)	6.02 (3.84)	1.33 (0.94)	0.55 (0.47)	5.32 (2.39)

n = 32 (Layout 1 to 4), n = 128 (Total)

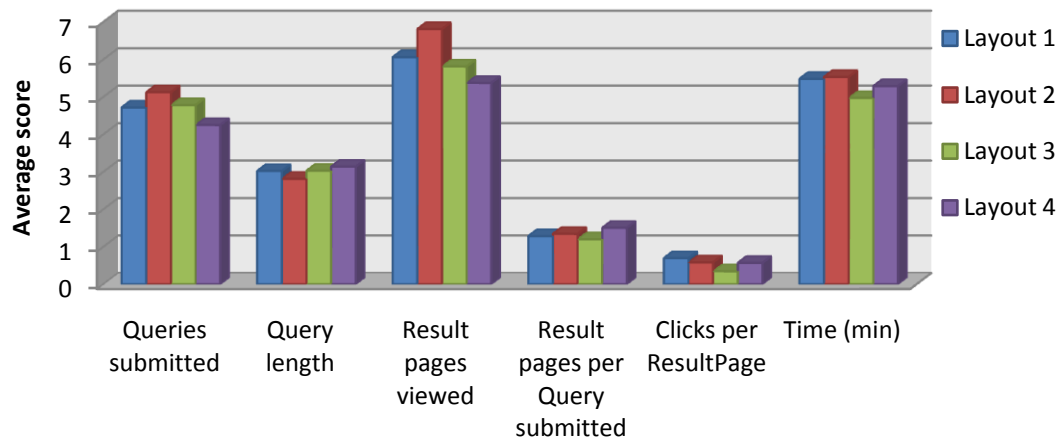


Figure 44: User Interaction

Discussion

Within the second column the amount of queries submitted per layout is least for the fourth layout; this indicates that the users needed to submit fewer queries to obtain the necessary information when using the fourth layout. What is peculiar though is that for the first and third layout more or less the same amount of queries were submitted, while the most queries were submitted when using layout two.

The query length column indicates that longer queries were submitted when using the fourth layout so the users formulated more specific queries when using this layout, this could suggest that they used the extra information available to reformulate more specific queries.

The users viewed the least amount of result pages when using the fourth layout, suggesting less result pages were needed to find what they were looking for. Layout 3 shows the least “result pages viewed per query submitted” while layout 4 shows the most.

The most documents were clicked when using the first layout; this indicates that the information provided in the baseline was not sufficient so more often they had to open a document to find out if this was relevant for their search. This was also clear during the user evaluation that users clicked fewer documents when using the layouts 2-4 because it offered enough information to complete certain tasks. Another interesting factor is that seeing that the underlying search engine is identical for all layouts, and that the users clicked less documents per result page when using layouts 2-4. It can be suggested that the participants interacted with the system more often when using the document

summary layouts then when using the baseline layout. So TRS and thumbnails appear to increase the interaction between the user and search engine.

From these results nothing concrete can be concluded because the standard deviations are all quite large, meaning that the results are spread broadly over the result space. Also the differences between the average values between the layouts are not very big, these only suggest slight differences. This may also be because of the assignments used; it is very difficult to set up assignments, which actually give a good representation of reality.

8.2.2 Relevance Assessment

This paragraph presents the participants relevance assessment from three perspectives as shown in the table below from the left to the right column respectively: “Ease of finding information using the current layout”, “Easy of finding new Information with the current layout after query reformulation” and “Easy with which the user could predict the contents of the documents based on the information available in the layout”.

The second perspective namely “New Information” will be elaborated on further because it seemed to cause some confusion during the user evaluation. When a user is looking for some information and the first query used did not present the results desired and the user reformulates the query and submits this, does this actually present new information to the user or did the layout not contribute enough information for query reformulation.

For each of the perspectives the participants were asked to fill in a score, between 1 and 7, 1 indicating best/easiest and 7 worst/hardest. The values in each of the cells are the average and the standard deviation (between brackets).

Table 11: Relevance Assessment

Layout	Ease of Finding	New Information	Contents Prediction
1	2.72 (1.28)	3.28 (1.08)	2.41 (1.48)
2	2.72 (1.35)	2.88 (1.16)	2.28 (1.20)
3	1.97 (1.26)	2.69 (1.33)	2.03 (1.18)
4	1.78 (0.79)	2.59 (1.16)	1.78 (1.01)
Total	2.3 (1.20)	2.86 (1.20)	2.13 (1.24)

n = 32 (Layout 1 to 4), n = 128 (Total)

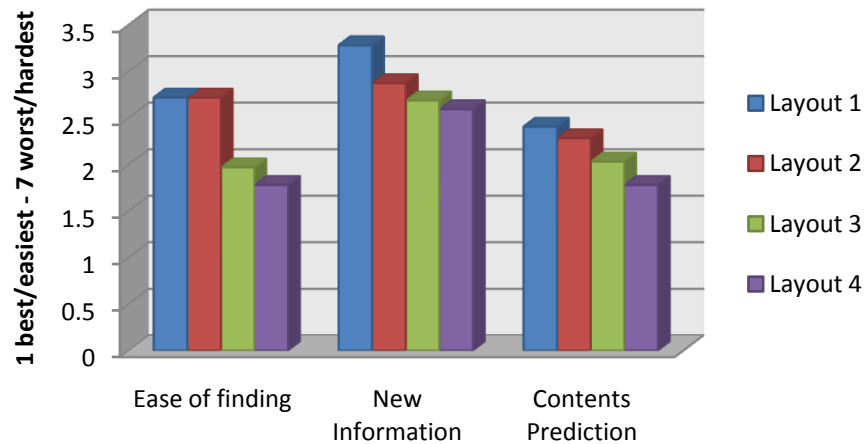


Figure 45: Relevance Assessment

Discussion

The ease of finding is greater using the third and fourth layout, this suggests that the thumbnail contributed the most in facilitating the ease of finding for the user. While for the baseline and second layout the value is the same, this indicates that the ease of finding with the second layout is equal to the baseline, so the TRS do not contribute to this measure.

The layouts 2, 3 and 4 are preferred over the baseline layout when reformulating a query to attain new information. There is a definite difference between the baseline and the document summary layouts, which indicates that these assist a user with query reformulation.

For contents prediction the results show that layout 4 is best followed by layout 3, then 2 and finally layout 1. This indicates that the contents prediction increases as the elements TRS and Thumbnails are added to the interface, this suggests that the users prediction accuracy increases as more information is available, which is quite a logical occurrence. This was also noted during the user evaluation that the users could complete several assignments faster, because the extra information allowed them to form better opinions on what a document contained.

For each of the perspectives the best rated overall is it the fourth layout, this suggests that this is the most optimal layout and maximizes the user's relevance assessment.

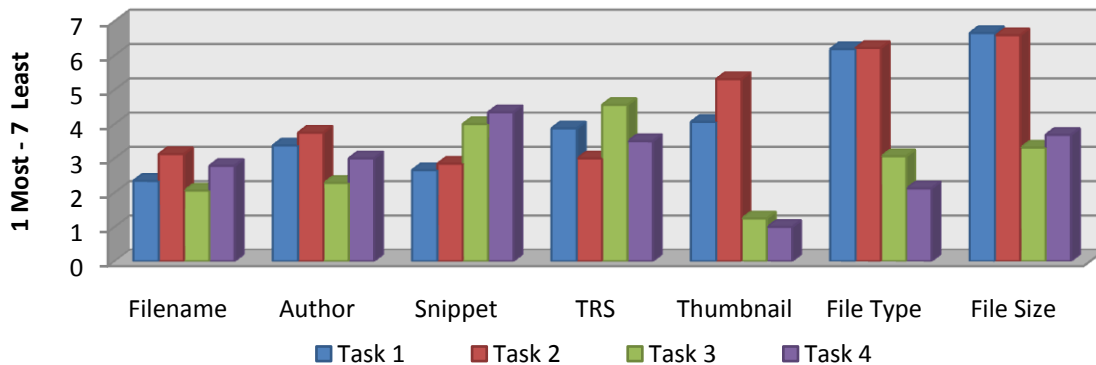
8.2.3 Contribution of Layout Elements

The previous sections showed the advantages of adding new elements to the baseline layout. These results indicate that layout 4 is more likely to offer better support in a users information seeking process than layout 1. In this paragraph the results obtained from the user tests concerning the contribution of the layout elements are presented. Each of the layout elements got a score from the participants that corroborated how much each of them contributed to the initial relevance assessment. The scores assigned were again between 1 and 7, 1 indicating very much and 7 very little. The following layout elements were evaluated: Filename, Author, Snippet, TRS, Thumbnail, File type and File size. The results are shown in Table 12 where a strong contribution is represented by a low score. The values in each of the cells are the average and the standard deviation (between brackets). Note that the sample sizes differ across the layout features seeing that the layout elements TRS and Thumbnail were not available in all the layouts.

Table 12: Contribution of layout elements

Task	Filename	Author	Snippet	TRS	Thumbnail	File type	File size
1	2.34 (1.64)	3.38 (1.88)	2.66 (1.21)	3.88 (1.63)	4.06 (1.61)	6.19 (1.40)	6.66 (1.10)
2	3.13 (1.72)	3.75 (2.08)	2.84 (1.92)	3.00 (1.41)	5.31 (1.99)	6.22 (1.39)	6.59 (1.13)
3	2.06 (1.72)	2.28 (1.65)	4.00 (2.00)	4.56 (1.46)	1.25 (0.45)	3.06 (2.05)	3.31 (2.35)
4	2.78 (2.01)	3.00 (1.97)	4.34 (1.86)	3.50 (1.41)	1.00 (0.00)	2.13 (1.56)	3.69 (2.46)
Total	2.58 (1.86)	3.1 (1.96)	3.46 (1.90)	3.73 (1.56)	2.91 (2.24)	4.40 (2.44)	5.06 (2.43)

n = 16 (for TRS and Thumbnail in Task 1 to 4), n = 32 (the rest in Task 1 to 4), n = 64 (for TRS and Thumbnail in Total), n = 128 (the rest in Total)

**Figure 46:** Contribution of layout elements

The tasks shown in Table 12 and Figure 46 correspond with the ones discussed in paragraph 7.4.5. Task 1 is a Background Information Task, while task 2 represents a Decision Making Task. Task 4 models a Known Item Task and the finally task 4 is a Topic Distillation Task.

Discussion

From the total row in Table 12 can be concluded that the participants found the filename was the strongest factor when determining whether or not to open a document. While the thumbnail comes second again demonstrating its importance as shown in paragraph 8.2.2. The difference between the TRS and the Snippet is not very significant but it does indicate that users tend to use the Snippet more. This could be because the participants are accustomed to the Snippet being a crucial element within the result presentation, and that they still have to get use to the TRS. This was also mentioned by several participants after the evaluation.

When the TRS and Thumbnail results are compared it is clear that TRS is given a stronger score for task 2 a decision making task, while the thumbnail is given a stronger score in task 4 a known item task. This suggests that the effectiveness of TRS and Thumbnails varies across tasks. And this also indicates that TRS may be more useful where Thumbnails are less effective, and vice versa.

Another interesting observation is that the Thumbnail is given a very high score for the tasks 3 and 4, while the TRS is given its strongest score for task 2. This can be explained cause for task 3 and 4 the user had to find documents of which the front page was known, so the thumbnail made finding the documents a breeze. While for task two the user had to search trough the dataset to figure out whether or not a project had been carried out or not, so for this task having extra information of the TRS was very useful.

For the file type it is also noticeable that for task 3 and 4 it scores very high because the users knew the file type of the documents they were looking for so the users could appreciate all the file type indicator elements within the user interface.

The file type and file size elements score very low for the first and second task cause these elements were not needed to complete these tasks. But there was one expert that did use these elements during these tasks; the reasoning behind this was that the larger the file the bigger the chance that it was a thesis work and finding these documents was essential to completing the first and second task.

A very interesting observation is that the author element was not rated very high for the first task where the objective was to find the name of the author corresponding to a certain project and vice versa. This was mainly because the author field information is extracted from the metadata that is gathered from the document and this is not always completely accurate. Sometimes it contains nothing or a useless variable. It could be suggested that a better means of acquiring the information stored in the metadata should be considered, or the use of metadata should be encouraged or facilitated. This could be done by allowing a Content Management System (CMS) to fill in the information about the author; seeing that within a company CMS this metadata is usually readily available. Some more of the information that can be stored in the metadata is Title of the document, Subject, Keywords, Comments, Category and Revision number. The last one can be updated by a CMS as well; the rest of these are more difficult to extract automatically from the document. Metadata is a helpful tool to easily extract useful information from a document which can be indexed by a Search Engine. But research should be done how to ensure this data is always available, or to explore different means of automatically extracting this information. This information can improve the index of a Search Engine which of course will improve the chances of a searcher actually finding its target information.

8.2.4 Layout Preference

Once the participants had completed all the tasks they were asked to rank all the layouts according to their preference. The scores 1 to 4 were used, 1 being the one preferred most and 4 least. In the figure and table below the results are presented. The clear winner among the layouts is layout four with an average of 1.5 followed by layout 3 as the runner up. Twenty of the thirty-two participants choose layout 4 as the preferred layout, while twenty-three participants choose layout 1 as the least preferred one. Some of the users did not rank layout 4 as the winner because they think it is a bit too cluttered, and they are used to the current way of presenting search results. This suggests that the new layouts take some getting used to.

Table 13: Layout preference

Preference	Layout 1	Layout 2	Layout 3	Layout 4
1 (Most)	0	1	11	20
2	6	4	13	9
3	3	20	7	2
4 (Least)	23	7	1	1
Average rank	3.53	3.03	1.94	1.50

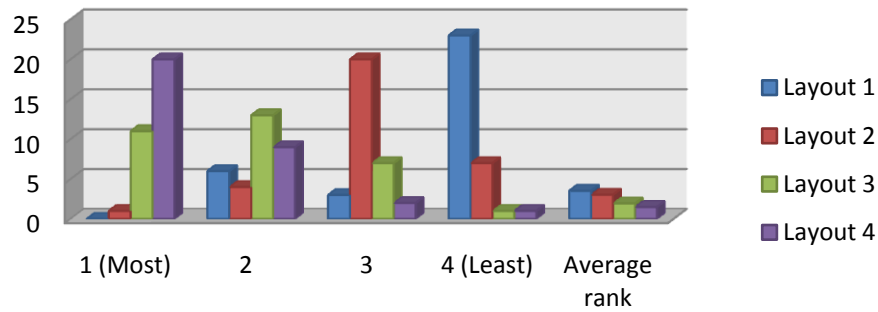


Figure 47: Layout preference

An interesting observation was that layout preference is dependent on the type of search task that needs to be completed. If a user is looking for a PowerPoint presentation and they remember the image on the front page of the presentation it is very handy if there is a thumbnail available. The overall positive performance of layout 4 could also be that it offers a support in a wider range of tasks that layout 2 or 3.

8.2.5 Participant Comments

This paragraph briefly gives an overall view of the comments made by the participants. One of the most noticeable comments was that TRS were experienced as being very handy, but the users did have to get use to the idea of them. Some users also thought this was a redundant element, others thought that maybe the TRS should replace the snippet. The thumbnails seemed to be the participants preferred addition; this indicates that this element can be very useful when performing a particular type of search. Some comments were made about the design of the interface, mainly that some of the fonts should be a bit bigger, but this is a usability issue which can always be tweaked. The prototype's sorting and filtering feature was also a hit among the participants.

One of the experts implied that it was very obvious how for each of the evaluation tasks an optimal layout exists, showing that maybe a user should be able to choose a layout based on the search task that has to be carried out. Another expert pointed out that the TRS is a practical element when looking for content while a thumbnail is redundant.

This was a general overview of the comments made by the participants after the user evaluation.

Overall the opinions on the prototype were very positive.

9 Conclusions and Recommendations

This chapter focuses on summarizing the work that was done during the course of this research. Hereafter some concluding remarks are presented, followed by the implications this work has for the design of current Enterprise Search systems. Lastly some future research topics are discussed.

Research Phase

The general focus of this thesis work was to “Design, implement and evaluate a system for Enterprise Information Retrieval, which employs an improved search result presentation technique”. This was done by first carrying out a literary study of current progress in Information Retrieval. Next current Enterprise Search systems were explored; hereafter research was done on search result presentation techniques. This research led to finding two forms of document summaries, namely a textual document summary technique: Top Ranking Sentences (TRS) and a visual document summary technique: Thumbnails (discussed in paragraph 3.1). TRS are up to three sentences presented in the search results that contain the query terms and are ranked according to several features. Their main purpose is to offer the user extra information about a document, so beforehand they can form a better opinion of what information a document contains [P3]. The thumbnail is a miniature image of the first page of a document; this also increases ability of a user to make more accurate decisions about the relevance of search results [P4]. These document summaries have been designed to support a user during information seeking activity. Hereafter a cognitive model for Information Retrieval was designed, which was used to get a better understanding of the information seeking process. This was necessary for designing and developing the new system for Enterprise Information Retrieval.

Design & Development Phase

After this research, a design was forged for the Basora Enterprise Search (BES) prototype using the knowledge acquired; its main purpose is to make a dataset searchable while assisting a user with query reformulation and relevance assessment. This design consisted of various components namely Crawler, Parser, Committer, Indexer & Searcher, Basora Engine and a Web Interface. These were implemented using an agile software development approach, incrementally developing BES prototype. The Crawling Parsing and Committer components are the pre-processing steps of the BES system. These were developed from scratch using some essential libraries namely Apache POI & PDFBox (for reading PDF and Office documents). The pre-processing components are responsible for retrieving all documents from a file system, extracting all the metadata and content, formatting this into indexable XML and finally committing this to the Indexer. The Apache Solr Server (an open source Search Server 5.2.2) was customized to provide Indexer and Searcher support for the BES system. Its main function is ensuring that the pre-processed data is properly stored, and it also needs to retrieve the results as quickly as possible once a query is submitted to the Searcher. The Basora Engine’s main purpose is to allow the Web Interface and the Solr Searcher to communicate. It is also responsible for extracting the TRS, and formatting the results in HTML so they can be displayed to the user through the Web Interface. This interface allows users to interact with the BES system during the extent of their search. A custom TRS Algorithm was developed using several papers as a guideline (presented in paragraph 5.2.5 and 6.2.6). The BES prototype contains 4 different layouts (discussed in paragraph 6.2.5) for search result presentation. The first contains the same amount of information as the current Enterprise Search Solutions provide to their users, this one is referred to as the baseline. While the other three layouts consist of the baseline components with the addition of either or both, the visual and/or textual document summaries (TRS and Thumbnail). The BES system was developed using the Java 1.6 programming language, Java EE, HTML 4.01, and CSS 2.0.

Evaluation Phase

Then the prototype was put through a performance and user test. The performance test (paragraph 8.1) was carried out by measuring the systems pre-processing time, response time and amount of results obtained per user query. This was done using 30 queries for both the BES prototype and the IBM Omnifind 8.5 (used as a benchmark) search solution.

The performance test indicated that BES and Omnifind systems pre-processing time is more or less equal. BES's average response time was only 75 milliseconds longer than the IBM Omnifind, while providing the user with two extra forms of document summaries. Collins 1994 [B3] states that a user considers a process to be real time if it takes less than 100 milliseconds. So this additional time is not noticeable by users and won't become an annoyance to searchers. Additionally the BES system returns about 11 times more search results than the IBM Omnifind system; this could be because the Omnifind engine has a stricter filtering function, but it could also indicate that the Omnifind engine indexes less of the documents contents than the Basora prototype does.

Subsequently the user tests (paragraph 8.2) were carried out according to the IMPACT model for user evaluation; to evaluate whether the BES prototype's new search result presentation technique, actually improves a users relevance assessment and ability to reformulate search queries. The user tests were carried out with 32 participants. They each performed four assignments (Appendix F: Evaluation Assignment Form); every assignment was done with a different search result presentation layout (presented in paragraph 6.2.5).

The results of the user test demonstrate that there are various cases where the addition of these textual and visual summary elements (TRS and Thumbnail 6.2.5) has a positive effect on relevance assessment and query reformulation. What can also be suggested is that the effectiveness of TRS and Thumbnails varies across tasks, so that TRS may be more useful where Thumbnails are less effective and vice versa. Another interesting point is that the user evaluation also indicated that the participants often found it easier to find relevant documents and new information when TRS and Thumbnail elements were added to the interface. This suggests that the current search engine's result presentation technique is not necessarily optimised and that there is room for improvement. This evaluation procedure investigated a wider range of aspects of information seeking behaviour than previously carried out within Enterprise Search. From this user evaluation could be concluded that the fourth layout¹⁴ is the most optimal search result presentation layout, not only because the participants selected it as the preferred layout, but it also offers support for a wider range of search tasks than any of the other layouts.

Concluding Remarks

This research demonstrates that this new Enterprise Information Retrieval system, Basora Enterprise Search actually helps a user assess the relevance of a document minimizing the amount of documents that need to be opened before the user finds the desired one. It also suggests that the additional visual and textual document summaries assist the user when reformulating a search query, also decreasing the time it will take a user to complete a search task. Finally it also indicates that it is possible to build a system that performs more or less equal to the current response time standards of Enterprise Search Engines, containing these useful components.

9.1 Implications

The results from this research project have several implications for the design of Enterprise Search Engine interfaces. Especially because they demonstrate that the addition of these textual and visual

¹⁴ Layout 4: Baseline, TRS and Thumbnail (shown in paragraph 6.2.5)

document summary elements has a positive effect on relevance assessment and query reformulation. Also the fact that the system performs more or less equal to the current response time standards of Enterprise Search Engines, makes it attractive to consider adding this improved search result visualization technique to current Enterprise Search Interfaces. Another point that also supports the previous statement is that the participants of the user evaluation often found it easier to find relevant documents and new information when TRS and Thumbnail elements were added to the search result interface so this suggests that the current Enterprise Search engine's result presentation technique is not necessarily optimised and that there is room for improvement.

The BES system is not completely optimized, especially the pre-processing steps parsing component because this relies on open source Java libraries that are work in progress. Another minor issue is that support should also be built in for the new Office document format, seeing that its popularity is increasing. These are some of the aspects that still need optimization but even so the system demonstrates that it can improve a user's search experience.

It is not the intention of this research project to suggest that the current interfaces are useless, because this is surely not the case seeing that these have been used years and they are effective. But the intention is to show that the evolutionary cycle of Enterprise Search Engine Interface design, has reached a new stage. Serious considerations should be made in favour of this advance, because the addition of these document summaries not only facilitates query reformulation and relevance assessment. But also increases the level of interaction between the user and the search engine interface; also assisting in closing a gap the science of Human Computer Interaction.

9.2 Future Research

This paragraph presents some areas where extended research could still be carried out. These suggestions can be carried out using the current prototype with some minor modifications. It also demonstrates the diverse re-use capabilities of the developed Enterprise Search prototype.

Multiple Thumbnails

Add an option for choosing thumbnails of multiple document pages, and researching whether this would increase the relevance assessment of users. The reason for this is that a lot of documents within a company have similar front pages but the rest of the pages are usually unique. Examples could be the page with the table of contents or the introduction.

Retrieval Algorithm

Research whether the actual retrieval algorithm of a search engine could incorporate the Top Ranking Sentences principle during the retrieval process. This would mean a different way of indexing in order to calculate the TRS values as quickly and efficiently as possible.

User Centred Design

Another interesting research assignment could be executing a full user centred design process, to obtain the rest of the bottlenecks in the current Enterprise Search engine interface. This could be used to design and develop better Enterprise Search engine interfaces.

Metadata

Research the possibility of either facilitating the process of filling in metadata for a user, or automatically extracting this data from a document. Metadata can play an important role in assisting a user find its target information. This information is usually not available because a user is not required

to fill this in, and it is difficult to do this manually. Research in this area can also improve the accuracy of current Search Engines.

Enterprise Parsing

The Basora Enterprise Search (BES) prototypes pre-processing components have been developed to be robust. But seeing that the development of these components was not the main focus of this research they have not been fully optimized. A component that still needs to be optimized is the parser, which is highly reliant on the PDFBox and Apache POI Java libraries. These are all open source libraries and do still contain several bugs. This of course also affects the BES parser, so the further development and optimization of the parser could also be a research assignment on its own.

Bibliography

Books

- [B1] Benyon, D. and Turner, P. and Turner, S. 2005, *“Designing Interactive Systems”*, 1st edition, Harlow Essex England. 2005.
- [B2] Langville, A.N. and Meyer, C.D. 2006, *“Google’s PageRank and Beyond: The Science of Search Engine Rankings”*, 1st edition, Princeton University Press, Woodstock Oxfordshire England, 2006.
- [B3] Collins, D. 1994, *“Designing Object-Oriented User Interfaces”*, 1st edition, The Benjamin/Cummings Publishing Company Inc. Redwood City, California, 1994.

Research Assignment

- [RA] *Development in Information Retrieval*, M. Wallé, 2008.

Papers

- [P1] Frappaolo, C. and Keldsen, D. 2008, *“Findability: The Art and Science of Making Content Easy to Find”*, in Market IQ Intelligence Quarterly Q2 2008, AIIM, 2008.
- [P2] Joho, H. and Jose, J.M. 2006. *“A Comparative Study of the Effectiveness of Search Result Presentation on the Web”*, In: Proceedings of the 28th European Conference on Information Retrieval, volume 3936 of Lecture Notes in Computer Science, London, UK, April 10-12, 2006, pages 302-313, Springer-Verslag, 2006.
- [P3] White, R.W. and Jose, J.M. and Ruthven, I. 2005, *“Using Top-Ranking Sentences to Facilitate Effective Information Access”*, Journal of the American Society for Information Science and Technology, 2005.
- [P4] Dziadosz, S. and Chandrasekar, R. 2002, *“Do Thumbnail Previews Help Users Make Better Relevance Decisions About Web Search Results?”*, in proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval , 2002.
- [P5] Brin, S. and Page, L. 1998. *“The anatomy of a large-scale hypertextual Web search engine”*, in journal of Computer Networks and ISDN Systems, Elsevier, 1998
- [P6] Borlund, P. 2000. *“Experimental components for the evaluation of interactive information retrieval systems”*, In: Journal of Documentation, MCB UP Ltd, Volume 56, Issue 1, pages 71-90, 2000.
- [P7] Belkin, N.J. and Oddy, R.N. and Brooks, H.M. 1982, *“ASK for information retrieval: Part I. Background and theory”*. Journal of Documentation, 38(2): pages 61-71, 1982.

- [P8] Belew, R. 2000, *“Finding Out About – Search Engine Technology from a Cognitive Perspective”*, Cambridge University Press, 2000.
- [P9] Hearst, M.A. and Pederson, J.O. 1996, *“Re-examining the Cluster Hypothesis: Scatter/Gather on Retrieval Results”*, in Proceedings of the 19th annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Zurich, Switzerland: ACM 1996.
- [P10] Jansen, B.J. et al 1998, *“Real Life Information Retrieval: A Study of User Queries on the Web”*, ACM SIGIR Forum: A Publication of the Special Interest Group on Information Retrieval, 32(1): pages 5-17, 1998.
- [P11] Enterprise Search Evaluation Guide, 2006, *“Seven critical characteristics your enterprise search solution must have”*, 2006. [<http://www.safira.pt/NR/rdonlyres/C506ED0A-301C-40E9-8E41-57A2E8ED3C20/97/EnterpriseSearchEvaluationGuide.pdf>], last visited: 20 October 2008.
- [P12] Tombros, A. and Ruthven, I. 1998, *“Advantages of query-biased summaries in information retrieval”*, in Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Melbourne, Australia: ACM. 1998.
- [P13] Wang, J., Vries de, A.P., and Reinders, M.J.T. 2006. *“A user-item relevance model for log-based collaborative filtering”*. In: Proceedings of the 28th European Conference on Information Retrieval, volume 3936 of Lecture Notes in Computer Science, London, UK, April 10-12, 2006, pages 37-48, Springer-Verslag, 2006.
- [P14] McDonald, R. 2007. *“A Study of Global Inference Algorithms in Multi-document Summarization”*, In: Proceedings of the 29th European Conference on Information Retrieval Research, volume 4425 of Lecture Notes in Computer Science, Rome, Italy, April 2-5, 2007, pages 557-564, Springer-Verslag, 2007.
- [P15] Demartini, G. and Mizzaro, S. 2006. *“A Classification of IR Effectiveness Metrics”*, In: Proceedings of the 28th European Conference on Information Retrieval, volume 3936 of Lecture Notes in Computer Science, London, UK, April 10-12, 2006, pages 488-491, Springer-Verslag, 2006.
- [P16] Mooney, C., Scully M., Jones G.J.F., and Smeaton, A.F. 2006. *“Investigating Biometric Response for Information Retrieval Applications”*, In: Proceedings of the 28th European Conference on Information Retrieval, volume 3936 of Lecture Notes in Computer Science, London, UK, April 10-12, 2006, pages 570-574, Springer-Verslag, 2006.
- [P17] Chernov, S., Serdyukov, P., Chirta, P.A., Demartini, G. and Nejd, W. 2007. *“Building a desktop Search Test-Bed”*, In: Proceedings of the 29th European Conference on Information Retrieval Research, volume 4425 of Lecture Notes in Computer Science, Rome, Italy, April 2-5, 2007, pages 686-690, Springer-Verslag, 2007.
- [P18] Yeung, P.C.K., Buttcher, S., Clarke, C.L.A. and Kolla, M. 2007. *“A Bayesian Approach for Learning Document Type Relevance”*, In: Proceedings of the 29th European Conference on Information Retrieval Research, volume 4425 of Lecture Notes in Computer Science, Rome, Italy, April 2-5, 2007, pages 753-756, Springer-Verslag, 2007.

- [P19] Yao, J. Wang, Z. Li, M. Li, and W.Y. Ma, 2006. “*Ranking Web news via homepage visual layout and cross-site voting*”, In: Proceedings of the 28th European Conference on Information Retrieval, volume 3936 of Lecture Notes in Computer Science, London, UK, April 10-12, 2006, pages 131-142, Springer-Verslag, 2006.
- [P20] Shah, C. and Eguchi, K. 2007. “*Use of Topicality and Information Measures to Improve Document Representation for Story Link Detection*”, In: Proceedings of the 29th European Conference on Information Retrieval Research, volume 4425 of Lecture Notes in Computer Science, Rome, Italy, April 2-5, 2007, pages 393-404, Springer-Verslag, 2007.
- [P21] Bogers, T. and van den Bosch, A. 2006. “*Authoritative Re-ranking of Search Results*”, In: Proceedings of the 28th European Conference on Information Retrieval, volume 3936 of Lecture Notes in Computer Science, London, UK, April 10-12, 2006, pages 519-522, Springer-Verslag, 2006.
- [P22] Vildjiounaite, E. and Kallio, S. 2007. “*A Layered Approach to Context-Dependent User Modelling*”, In: Proceedings of the 29th European Conference on Information Retrieval Research, volume 4425 of Lecture Notes in Computer Science, Rome, Italy, April 2-5, 2007, pages 749-752, Springer-Verslag, 2007.
- [P23] Osinski, S. 2006. “*Improving Quality of Search Results Clustering with Approximate Matrix Factorisations*”, In: Proceedings of the 28th European Conference on Information Retrieval, volume 3936 of Lecture Notes in Computer Science, London, UK, April 10-12, 2006, pages 167-178, Springer-Verslag, 2006.
- [P24] Sevillano, X., Cobo, G., Alías, F. and Socoró, J.C. 2007. “*A Hierarchical Consensus Architecture for Robust Document Clustering*”, In: Proceedings of the 29th European Conference on Information Retrieval Research, volume 4425 of Lecture Notes in Computer Science, Rome, Italy, April 2-5, 2007, pages 747-744, Springer-Verslag, 2007.
- [P25] Yamout, F., Oakes, M. and Tait, J. 2006. “*Relevance Feedback Using Weight Propagation*”, In: Proceedings of the 28th European Conference on Information Retrieval, volume 3936 of Lecture Notes in Computer Science, London, UK, April 10-12, 2006, pages 575-578, Springer-Verslag, 2006.
- [P26] Roulland, F., Kaplan, A., Castellani, S., Roux, C., Grasso, A., Pettersson K. and O'Neill, J. 2007. “*Query Reformulation and Refinement Using NLP-Based Sentence Clustering*”, In: Proceedings of the 29th European Conference on Information Retrieval Research, volume 4425 of Lecture Notes in Computer Science, Rome, Italy, April 2-5, 2007, pages 210-221, Springer-Verslag, 2007.
- [P27] Lalmas, M., Rüger, S., Tsirikika, T. and Yavlinsky, A. 2006. “*Progress in Information Retrieval*”, In: Proceedings of the 28th European Conference on Information Retrieval, volume 3936 of Lecture Notes in Computer Science, London, UK, April 10-12, 2006, pages 1-11, Springer-Verslag, 2006.
- [P28] Peng, J. and Ounis, I. 2007. “*Combination of Document Priors in Web Information Retrieval*”, In: Proceedings of the 29th European Conference on Information Retrieval Research, volume

4425 of Lecture Notes in Computer Science, Rome, Italy, April 2-5, 2007, pages 732-736, Springer-Verslag, 2007.

- [P29] Wang, J. and Roelleke, T. 2006. *“Context-Specific Frequencies and Discriminateness for the Retrieval of Structured Documents”*, In: Proceedings of the 28th European Conference on Information Retrieval, volume 3936 of Lecture Notes in Computer Science, London, UK, April 10-12, 2006, pages 579-582, Springer-Verslag, 2006.
- [P30] Cornacchia, R. and de Vries, A.P. 2007. *“A Parameterized Search System”*, In: Proceedings of the 29th European Conference on Information Retrieval Research, volume 4425 of Lecture Notes in Computer Science, Rome, Italy, April 2-5, 2007, pages 4-15, Springer-Verslag, 2007.
- [P31] Chen, C., Oakes, M. and Tait, J. 2006. *“Browsing personal images using episodic memory (time + location)”*, In: Proceedings of the 28th European Conference on Information Retrieval, volume 3936 of Lecture Notes in Computer Science, London, UK, April 10-12, 2006, pages 362-372, Springer-Verslag, 2006.
- [P32] Boyer, A. and Brun, A. 2007. *“Natural Language Processing for Usage Based Indexing of Web Resources”*, In: Proceedings of the 29th European Conference on Information Retrieval Research, volume 4425 of Lecture Notes in Computer Science, Rome, Italy, April 2-5, 2007, pages 517-524, Springer-Verslag, 2007.
- [P32] Boyer, A. and Brun, A. 2007. *“Natural Language Processing for Usage Based Indexing of Web Resources”*, In: Proceedings of the 29th European Conference on Information Retrieval Research, volume 4425 of Lecture Notes in Computer Science, Rome, Italy, April 2-5, 2007, pages 517-524, Springer-Verslag, 2007.
- [P33] Edmundson, H. and Wyllys, R. 1961. *“Automatic abstracting and indexing---survey and recommendations”*, In: Communication of the ACM, New York USA, volume 4, issue 5, pages 226-234, 1961.
- [P34] Saggion, H. 1999. *“Using linguistic knowledge in automatic abstracting”*, In: Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics, Morristown New York USA, pages 596-601, 1999.

Appendix

Appendix A: Update Meeting Sheet

Persons present

- ∴ E. Essenius
- ∴ B. Vranken
- ∴ M. Wallé

Introduction

This document gives an overview of the user stories (Implementation requirements), that have been completed and also the ones that have yet to be implemented.

Hurdles

The hurdles left in order to finalize the prototype are (these are sorted by relevance):

- Finalize the overall Basora Enterprise search engine
 - Finish fixing some minor interface requirements
- Extract thumbnail from DOC, DOCX, XLS, XLSX, PPTX files.
 - Can be resolved either finding a java library for these purposes or converting these files to PDF and per curing the thumbnail using the PDF thumbnail extractor method.
- Extract data from XLSX, DOCX, PPTX files
 - Can be fixed by converting these to PDF.

Hurdles than have been overcome

These hurdles have been overcome during the last update meeting on the 7th of July.

- Extracting the Baseline and Top Ranking sentences
 - Baseline is now being extracted using the highlighter function of the Solr engine and a basic function that extracts this and displays it.
 - The Top Ranking sentences are now extracted using Solr highlighters to gather sentences containing the query terms, then a function calculates which are the top 3 sentences that need to be presented
- Crawl and retrieve data from WT workspace
 - It is now possible to crawl the WT workspace and retrieve all office and PDF files

Planning

Week	Activity	Period
16 - 25	Develop Prototype	9 weeks
25	Final test and debug session & Stress/Performance Test	1 week
28	Update Thesis and review draft	3 days
28 - 31	Evaluate prototype – Create questionnaire, execute experiment	4 weeks
32	Review and process all data retrieved from experiment	1 week
33 - 35	Update and evaluate Thesis	3 weeks
36 - 37	Additional Time	2 weeks
37	Present project at working tomorrow before final presentation	1 day
38 - 39	Time to prepare for final thesis defence and presentation	2 weeks

Gantt chart

Week	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
	21 Jul ~ 27 Jul	28 Jul ~ 3 Aug	4 Aug ~ 10 Aug	11 Aug ~ 17 Aug	18 Aug ~ 24 Aug	25 Aug ~ 31 Aug	1 Sep ~ 7 Sep	8 Sep ~ 14 Sep	15 Sep ~ 21 Sep	22 Sep ~ 28 Sep	29 Sep ~ 5 Oct	6 Oct ~ 12 Oct	13 Oct ~ 19 Oct	20 Oct ~ 26 Oct	27 Oct ~ 31 Oct
8	Develop prototype														
9	Final Test and Debug Session														
10	Update Thesis and review draft														
11	Evaluate prototype														
12	Review data retrieved from experiment														
13	Update and evaluate thesis														
14	Additional time														
15	Present project at working tomorrow before final														
16	Prepare for final thesis defense and presentation														

- The dark grey bar indicates the weeks that I am on vacation.

The planning shows that this week the development phase will end. I am planning on finalizing and tweaking the system during the coming two weeks so it is completely ready for its Stress test and its User tests.

Prototype Progress

This week's progress is shown in red.

Crawler

- ∴ Basic crawler [Crawls through windows file system and retrieves desired files]
 - Has been implemented ✓
- ∴ Advanced Crawler [Crawls through a workspace and retrieves all desired files]
 - Still has to be implemented

Parser

- ∴ Office document Parser [Apache POI package]
 - Has been configured ✓
 - Function for extracting all text from office documents ✓
 - Function for extracting all headings
- ∴ PDF Document Parser [PDFBox]
 - Has been configured ✓
 - Function for extracting all text from PDF documents ✓
 - Function for extracting all headings

Result Data Manipulator

- ∴ Develop Top Ranking Sentence (TRS) Extractor
 - Has not been implemented yet
- ∴ Develop Baseline Extractor
 - Has not been implemented yet
- ∴ Develop Office and PDF Thumbnail Extractor
 - PowerPoint thumbnail extractor is functional ✓
 - Word, Excel thumbnail extractor still has to be implemented
 - PDF thumbnail extractor still has to be implemented
- ∴ Develop Metadata Extractor
 - Extract Author ✓
 - Extract Title ✓
 - Extract File Size ✓
 - Extract Date Modified ✓
 - Extract Date Created ✓
 - Extract Page Count ✓
 - Extract Word Count ✓
 - Extract Keywords ✓
 - Extract Thumbnail from PPT ✓
 - Extract Thumbnail from PDF ✓
 - Extract Thumbnail from DOC
 - Extract Thumbnail from XLS
- ∴ Convert Office and PDF docs to suitable XML for indexing in SOLR
 - Convert Metadata to XML ✓
 - Convert Document Content to XML Office Documents ✓

- Convert Document Content to XML PDF Documents ✓

Indexer and Searcher

- ∴ Apache Solr package [Search engine core and searcher]
 - Is working for XML and CSV files ✓

Connection between Solr Searcher and Web Interface

- ∴ XML Parser
 - Custom Made Parser for high-speed performance ✓
- ∴ Connector
 - Read Solr Results and Parse these ✓
 - Feed the results to the result presentation functions ✓

Logging

- ∴ Log crawler progress
 - Log what document has currently been found and is written to crawled-data file ✓
 - Log if any errors occur so this can be fixed ✓
- ∴ Log parser progress
 - Log what document is being parsed and if it is parsed successfully ✓
 - Log if a document cannot be parsed ✓
 - Log if the thumbnail cannot be extracted from a document so this can be fixed
- ∴ Log post-index progress
 - Log what document is currently being posted to the SOLR index ✓
 - Log if it has been successfully posted ✓
 - Log all files that could not be posted ✓
- ∴ Log engine progress
 - Log all Queries submitted so these can be used as auto-complete data

Result Extraction

- ∴ Baseline Extraction
 - Extract baseline from results returned from Solr server ✓
- ∴ Top ranking sentences Extraction
 - Extract top ranking sentences from results returned from Solr server ✓

Web Interface

- ∴ Front web interface
 - First version is done ✓
 - Second version is done ✓
- ∴ Results web interface
 - Java functions have been implemented for basic HTML Generation ✓
 - Template Result Interface for Baseline, TRS & Thumbnail in HTML ✓
 - CSS styles have been defined ✓
 - Java Objects for containing and manipulating data have been created ✓
 - Generation of Result Interface by Servlet (Baseline, TRS & Thumbnail) ✓
 - Template Result Interface for Baseline in HTML ✓

- Template Result Interface for Baseline & TRS in HTML ✓
 - Template Result Interface for Baseline & Thumbnail in HTML ✓
 - Generation of Result Interface by Servlet (Baseline) ✓
 - Generation of Result Interface by Servlet (Baseline & TRS) ✓
 - Generation of Result Interface by Servlet (Baseline & Thumbnail) ✓
 - Options Menu – Choice of result presentation mode ✓
 - Options Menu – Choice of number of results per page ✓
 - Options Menu – Sort by [Relevance, Name, Date, Author, File size] option ✓
 - Options Menu – Sort order Ascending or Descending ✓
 - Sort by interface [Relevance, Name, Date, Author, File size] ✓
 - Sort by functionality [Relevance, Name, Date, Author, File size] ✓
 - Sort order interface [Ascending or Descending] ✓
 - Sort order functionality [Ascending or Descending] ✓
 - Filter results by document type interface ✓
 - Filter results by document type functionality ✓
- ∴ Crawler Interface
- Bases of interface ✓
 - Connect to crawler core ✓
- ∴ Parser Interface
- Bases of interface ✓
 - Connect to parser core
- ∴ Post-Index Interface
- Bases of interface ✓
 - Connect to post-index core

Data

- ∴ The WT workspace office and PDF documents will be used to test the system.
- Crawling the WT workspace ✓
 - Retrieving all the documents and caching this ✓

Appendix B: Final Assessment Prototype

Final additions

The final additions made to the prototype are listed in this paragraph. The interface can now inform the user properly when there are no results available for a certain query. The option that enables a user to go to the next set of results has been debugged and now works properly. The sorting and filtering bugs have now been resolved. Another small bug arose during testing, the problem was that when double quotes are used in the search box that the system did not convert these to the escape characters for the browser to be able to properly handle this. This bug has also been resolved.

Future Additions

These future additions are all elements that can still be added to the engine as upgrades:

- Extract thumbnail from DOC, DOCX, XLS, XLSX, PPTX files.

- Can be resolved either finding a java library for these purposes or converting these files to PDF and per curing the thumbnail using the PDF thumbnail extractor method.
- Extract data from XLSX, DOCX, PPTX files
 - Can be fixed by converting these to PDF
- Log all Queries submitted so these can be used as auto-complete data
- Multilingual character support
- Adjust the TRS ranking algorithm
 - By researching what additional variables could be used
 - And how the current ranking algorithm can be optimized
- Multilanguage analyser for indexing and searching.
 - This would allow for better indexing of the data

Planning

Week	Activity	Period
16 - 25	Develop Prototype	9 weeks
25	Final test and debug session & Stress/Performance Test	1 week
28	Update Thesis and review draft	3 days
28 - 31	Evaluate prototype – Create questionnaire, execute experiment	4 weeks
32	Review and process all data retrieved from experiment	1 week
33 - 35	Update and evaluate Thesis	3 weeks
36 - 37	Additional Time	2 weeks
37	Present project at working tomorrow before final presentation	1 day
38 - 39	Time to prepare for final thesis defence and presentation	2 weeks

Gantt chart

Week	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
	27 Oct ~ 31 Oct	20 Oct ~ 26 Oct	13 Oct ~ 19 Oct	6 Oct ~ 12 Oct	29 Sep ~ 5 Oct	22 Sep ~ 28 Sep	15 Sep ~ 21 Sep	8 Sep ~ 14 Sep	1 Sep ~ 7 Sep	25 Aug ~ 31 Aug	18 Aug ~ 24 Aug	11 Aug ~ 17 Aug	4 Aug ~ 10 Aug	28 Jul ~ 3 Aug	21 Jul ~ 27 Jul
8	Develop prototype														
9	Final Test and Debug Session														
10	Update Thesis and review draft														
11	Evaluate prototype														
12	Review data retrieved from experiment														
13	Update and evaluate thesis														
14	Additional time														
15	Present project at working tomorrow before final														
16	Prepare for final thesis defense and presentation														

- The dark grey bar indicates the weeks that I am on vacation.

Final Prototype Implementation Details

Crawler

- ∴ Basic crawler [Crawls trough windows file system and retrieves desired files]
 - Has been implemented ✓
- ∴ Advanced Crawler [Crawls trough a workspace and retrieves all desired files]
 - HTML Based Crawler ✓

- Folder Structure Based Crawler ✓

Parser

- ∴ Office document Parser [Apache POI library]
 - Has been configured ✓
 - Function for extracting all text from office documents ✓
- ∴ PDF Document Parser [PDFBox]
 - Has been configured ✓
 - Function for extracting all text from pdf documents ✓

Metadata Extractor and Thumbnail Extractor

- ∴ Develop Office and PDF Thumbnail Extractor
 - PowerPoint thumbnail extractor is functional ✓
 - PDF thumbnail extractor is functional ✓
- ∴ Develop Metadata Extractor
 - Extract Author ✓
 - Extract Title ✓
 - Extract File Size ✓
 - Extract Date Modified ✓
 - Extract Date Created ✓
 - Extract Page Count ✓
 - Extract Word Count ✓
 - Extract Keywords ✓
 - Extract Thumbnail from PPT ✓
 - Extract Thumbnail from PDF ✓
- ∴ Convert Office and PDF docs to suitable XML for indexing in SOLR
 - Convert Metadata to XML ✓
 - Convert Document Content to XML Office Documents ✓
 - Convert Document Content to XML PDF Documents ✓

Indexer and Searcher

- ∴ Apache Solr package [Search engine core and searcher]
 - Is working for XML and csv files ✓

Connection between Solr Searcher and Web Interface

- ∴ XML Parser
 - Custom Made XML Parser for high-speed performance ✓
- ∴ Connector
 - Read Solr Results and Parse these ✓
 - Feed the results to the result presentation functions ✓

Logging

- ∴ Log crawler progress
 - Log what document has currently been found and is written to crawled-data file ✓
 - Log if any errors occur so this can be fixed ✓

- ∴ Log parser progress
 - Log what document is being parsed and if it is parsed successfully ✓
 - Log if a document cannot be parsed ✓
 - Log if the thumbnail cannot be extracted from a document so this can be fixed ✓
- ∴ Log post-index progress
 - Log what document is currently being posted to the SOLR index ✓
 - Log if it has been successfully posted ✓
 - Log all files that could not be posted ✓

Result Extraction

- ∴ Baseline Extraction
 - Extract baseline from results returned from Solr server ✓
- ∴ Top ranking sentences Extraction
 - Extract top ranking sentences from results returned from Solr server ✓

Web Interface

- ∴ Front web interface
 - First version is done ✓
 - Second version is done ✓
- ∴ Results web interface
 - Java functions have been implemented for basic HTML Generation ✓
 - Template Result Interface for Baseline, TRS & Thumbnail in HTML ✓
 - CSS styles have been defined ✓
 - Java Objects for containing and manipulating data have been created ✓
 - Generation of Result Interface by Servlet (Baseline, TRS & Thumbnail) ✓
 - Template Result Interface for Baseline in HTML ✓
 - Template Result Interface for Baseline & TRS in HTML ✓
 - Template Result Interface for Baseline & Thumbnail in HTML ✓
 - Generation of Result Interface by Servlet (Baseline) ✓
 - Generation of Result Interface by Servlet (Baseline & TRS) ✓
 - Generation of Result Interface by Servlet (Baseline & Thumbnail) ✓
 - Options Menu – Choice of result presentation mode ✓
 - Options Menu – Choice of number of results per page ✓
 - Options Menu – Sort by [Relevance, Name, Date, Author, File size] option ✓
 - Options Menu – Sort order Ascending or Descending ✓
 - Sort by interface [Relevance, Name, Date, Author, File size] ✓
 - Sort by functionality [Relevance, Name, Date, Author, File size] ✓
 - Sort order interface [Ascending or Descending] ✓
 - Sort order functionality [Ascending or Descending] ✓
 - Filter results by document type interface ✓
 - Filter results by document type functionality ✓
- ∴ Crawler Interface
 - Basic interface ✓
 - Connect to crawler core ✓

- ∴ Parser Interface
 - Basic interface ✓
- ∴ Post-Index Interface
 - Basic interface ✓

Data

- ∴ The WT workspace office and pdf documents will be used to test the system.
 - Crawling the WT workspace ✓
 - Retrieving all the documents and caching these ✓

Appendix C: Entry Questionnaire

This questionnaire has to be completed before the user starts the user tests.

First Name	
Last Name	
Occupation	
Age	
Sex	

Search Experience	Years of experience using Web Search
Search Experience	Years of experience using Enterprise Search
Search Engines*	

*Search engines: Here you should fill in the search engines you use most frequently (e.g. Google, Yahoo, Windows Live Search, WebCrawler, IBM Omnifind, Fast, etc).

Appendix D: Session Questionnaire

The following questions have to be filled in every time you complete an assignment.

Question 1

Each column header corresponds to a layout element that is visible in the layout with which you have completed the previous assignment. You can see this on the layout sheet.

How much did each of these elements contribute to finding the requested information?

Please fill in a score between 1 and 7 for each column in the following table, 1 corresponding with very much and 7 very little.

There is an example available which indicates what header element correspond to what interface layout element.

Layout	Assign	Filename	Author	Location	Snippet	TRS	Thumbnail	File type	File size
	1								
	2								
	3								
	4								

Question 2

Please fill in a score between 1 and 7 for each question in the following table, 1 corresponding with very easy/often and 7 very hard/seldom. Please read the explanations of the questions listed below first to ensure that you properly understand the essence of the question.

Explanations:

1. When looking for information or documents using the current layout is it easy to obtain this?
2. When you are looking for a certain piece of information is it easy to find new information on that topic using the current layout?
3. When you find a document with the current layout, does the actual document contain what you previously expected to find based on the information provided by the layout elements.

Question	Assign 1	Assign 2	Assign 3	Assign 4
1. How easy is it to find the relevant documents from the search results?				
2. How easy is it to find new information on a topic while using the current layout?				
3. How often did the document contain what you expected to find?				

Appendix E: Exit Questionnaire

Please fill in the following table, indicate in each cell which layout you preferred. You do this by entering 1, 2, 3 and 4 for each of the cells. The layout that you preferred most you give a 1 and the one you preferred the least you give a 4.

	Layout 1	Layout 2	Layout 3	Layout 4
Preference				

Here you get a chance to express yourself freely and give some feedback on the layouts and the experiment itself.

Comments on the layouts

Comments on the experiment

Appendix F: Evaluation Assignment Form

Introduction

Thank you for taking part in the evaluation of the Basora Enterprise Search Engine. Today you will have to complete four tasks. These tasks need to be executed with **care**, they have all been created based on the simulated work task approach. The main idea is to evaluate the prototype in the environment where it will actually be used and using people that will end up using it. The prototype that will be evaluated has four layouts for presenting the search results. The main focus of this experiment is to evaluate the effectiveness of these layouts.

First you will start by filling in an entry questionnaire. Then you will have to complete four assignments and for each assignment a different layout will be used. You will get a total of 15 minutes to complete each assignment, after you complete each one you will be asked to fill in a session questionnaire. At the end of the experiment you will also be requested to complete an exit questionnaire.

If you have any questions beforehand please ask them?

Assignment 1

Assignment 1a

Please find the title of the project, corresponding to the following authors using the Basora search engine and fill in the title on the evaluation sheet.

Author Names
<i>William Tanis</i>
<i>Remco Nederpel</i>
<i>Emil Verwoerd</i>

Assignment 1b

Please find the names of the authors, corresponding to the following projects titles, using the Basora search engine and fill the names in on the evaluation sheet.

Project title
<i>Optimized Wireless Sensor Network for Multiple Train Carriages</i>
<i>Intelligente Tafel</i>
<i>Ontwerprichtlijnen Chinees-Westerse beleggingssite</i>

Session Questionnaire

Please fill in the session questionnaire for assignment 1.

Assignment 2

In this assignment you will get two assignment descriptions, you will need to do the following. Use the search engine to find out if these descriptions are new from what has already been done at Working Tomorrow.

Please find out if the following assignment descriptions are not in conflict with the ones already done or currently being done at Working Tomorrow. Once you are done searching please write down the author name of the documents that contain information regarding the topic on the evaluation form, if you did not find any documents just write an X in the documents cell and add some comments if you wish.

Assignment 2a

Robot Navigatie.

Met deze opdracht ga je kennis verwerven over autonome navigatie door robots. Hiervoor moet onderzoek worden gedaan naar demogelijkheden van robot navigatie en de belangrijkste technieken die er op dit gebied zijn. Het bedrijf wil de toepasbaarheid van deze kennis aantoonbaar maken in een prototype. Het doel van het prototype is een innovatieve toepassing van autonome navigatie door robots te demonstreren.

Assignment 2b

Agility and Software Architecture

Agile software ontwikkeling is de laatste jaren in populariteit toegenomen. Met deze opdracht ga je uitzoeken op welke manier de ontwikkeling van een software achitectuur kan worden ingebed in een agile software ontwikkel proces. Je zal met deze opdracht hand in hand werken met de Logica guru's op dit gebied, en het resultaat zal een eerste stap zijn in het verbeteren van ontwikkelprocessen binen Logica.

Session Questionnaire

Please fill in the session questionnaire for assignment 2.

Assignment 3

You now get 2 documents presented to you, the objective of this assignment is to locate them using the search engine. Once you have found them please write their file name, authors name and file size on your evaluation form.

You are allowed to keep the document open and look back into the file to gather inspiration for query terms.

Session Questionnaire

Please fill in the session questionnaire for assignment 3.

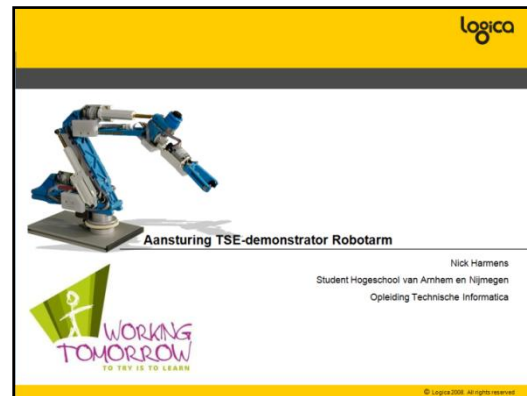
Assignment 4

Please locate the documents that are shown in the images below. Once you have found these please fill write down the documents file name, file size and the date it was created.

Assignment 4a



Assignment 4b



Assignment 4c



Session Questionnaire

Please fill in the session questionnaire for assignment 4.

Basora User Evaluation Form

Assignment 1a

Author	Project Title
William Tanis	
Remco Nederpel	
Emil Verwoerd	

Assignment 1b

Project Title	Author
Optimized Wireless Sensor Network for Multiple Train Carriages	
Intelligente Tafel	
Ontwerprichtlijnen Chinees-Westerse beleggingssite	

Assignment 2

Author names	Comments
2a.	
2b.	

Assignment 3

Document name	Author	File size (kb)
3a.		
3b.		

Assignment 4

Document name	File size (kb)	Date created
A.		
B.		
C.		

Appendix G: User Evaluation Procedure

1. Delete browser history
2. Fill in all the layout and iteration options on the questionnaire form before handing this to the user.
3. Let the user read the introduction form
4. Give the user the entry questionnaire –and allow them to read the session questionnaire beforehand.

- a. Answer the users questions if needed
 - b. Give the user the first assignment --- Change the interface layout accordingly
 - c. During the assignment write down how many pages are clicked per result page
 - d. When done: give the user the session questionnaire with the corresponding layout sheet
5. Give the user the second assignment --- Change the interface layout accordingly
 - a. When done: give the user the session questionnaire with the corresponding layout sheet
 - b. Make sure the person does not exceed 5-7 minutes completing the last assignment
 6. Give the user the third assignment --- Change the interface layout accordingly
 - a. When done: give the user the session questionnaire with the corresponding layout sheet
 7. Give the user the fourth assignment --- Change the interface layout accordingly
 - a. When done: give the user the session questionnaire with the corresponding layout sheet
 8. Give the user the exit questionnaire with the Layout Examples sheet
 9. Copy the users results to the results folder