Giggle:
Internet Search Engine for Kids

Jolanda van Gendt & Renske Weeda

IBM
Marketing Division
Supervisor: *Warner Dijkhuizen*

Radboud University Nijmegen
Information Retrieval and Information Systems
Supervisors: *dr. Franc Grootjen*
drs. ing. Pepijn Vos

Title:	Giggle, Internet Search Engine for Kids
Authors:	Jolanda van Gendt (026107)
	Renske Weeda (9913467)
Document type:	Master's thesis
Document number:	553
Date:	18th February 2005
Status:	Final
Supervisors:	Warner Dijkhuizen (IBM)
	Franc Grootjen (Radboud University Nijmegen)
	Pepijn Vos (Radboud University Nijmegen)
Keywords:	Information Retrieval, Children, Search Engine

Giggle: Internet Search Engine for Kids

by
Jolanda van Gendt and Renske Weeda

Master's thesis for Computer Science
Management & Technology variant

Nijmegen, the Netherlands

18th February 2005

Preface

A child searching for information on the Internet often attains results that are not meant for them; either too hard to read, written in the wrong language, or merely unappropriate to their age group (e.g. pornography, violence, racism). Also, current tools for Internet traversal do not adequately correspond to the capabilities nor the domain knowledge of a child.

In short, our investigation has lead us to the following conclusion. In order to support a child during their search on internet, they not only seek a search engine that will find information appropriate to their needs and interests, but they need a tool they feel comfortable with and attracted to, a tool that has been specifically built for children. The overall challenge that the user has during search is to specify a good query, the second is obtaining a manageable and relevant answer. Driven by these issues we have completed an extensive investigation, and now come with a breakthrough: a self-learning information retrieval system that meets the wishes and needs of our target group (children aged 8 to 12). This fully-automatic software system features a visual representation for navigating through cyberworld, aids in spell checking, expanding and tuning a search query, and yields results that beter adhere to a child's information need.

Driven by the issues involved with children on the internet, this thesis describes a proposal for a software system in an attempt to solve some of these problems. A prototype has been built to demonstrate and test the power of these ideas, including its social relevance. This thesis describes not only scientific research which forms the fundamental basis of the project, yet also describes the entire software engineering process, from requirements analysis, which serves as input for formal translation into design models and in turn into code. Final testing and evaluation has led to suggestions for improvements which can form the basis for future research. We have experiennced that such a systematic process is necessary in order to yield a qualitative software product.

Primarily this document has been written for our supervisors and our client to justify and document that the results and conclusions of our academic research are valid, precise, reproducible and integer. Secondly, it has been written for any person working on a similar project or problem, by describing the structured and systematic process with which we come to our results and the problems we ran into along the way.

Contents

Chapter 1

Introduction

Information is the key to the modern age, a new age that offers possibilities for the future limited only by the boundaries of our imaginations.

Tony Blair

1.1 Background

Giggle, Giechel, Risita. The project's deepest drive is to put a smile on your face, just as this project has put a smile on the faces of all of those involved throughout its existence.

"Giggle: Internet Search Engine for Kids" is a computer science and management and technology (MT) master's thesis project by two graduate computer science students at the Radboud University of Nijmegen, in assignment of IBM, Amsterdam. For more than six months, two university students, with the cooperation of many others, have made it possible to complete this project. Though motivation, goal and focus lay on different aspects for all those involved, this document forms a communication line between all parties, thereby communicating and describing the process and research accomplishments of the project and how different needs and wishes of those involved have been tied together, and furthermore, how these can be maintained and extended in the future.

It all started with IBM's involvement in the community with the intention to be a better corporate citizen than they already were, followed by an investigation in the community on which points this may be achieved. Our research had shown us that the community in which we live are striving to a safer environment for children on the Internet. Hand in hand with the children for which we built our product, inspired by their drive, their way of thinking and working, we worked on a system which would satisfy their needs, help them with their problems and yielding a system they could not only work with, but would also work with because it not only adheres to their needs, but is flexible to adhere to their wishes. Though the system was built to be used by children, we involved parents, teachers, librarians, governmental institutes and our clients (and their contacts) in determining the problems and possible solutions to the problems currently at hand. The result, obviously a lot of giggles and smiles! We were able to encounter, analyze, and document many needs and wishes with all stakeholders involved when it comes to the topic of children on internet. We composed creative ideas and devised and carried out a plan to fulfill those needs, and answer the desires and wishes encountered. What we present here is a prototype of an Internet Search Engine for Kids, ”Giggle”.

1.2 Current Environment

1.2.1 The Problem

A search engine is a piece of computer software that is used to search data for specified information. The most common are Internet search engines, particularly useful because the Internet is comprised of millions of documents in an unstructured matter, and finding data by just looking around is a rather impossible task. However, current search engines have their deficiencies.

Search engines are either too difficult to work with, or retrieve documents and information which are not relevant to the information need. Investigation into search patterns with adults has shown that most users conduct multiple searches, on average 22% of all searches return no results and the average success rate in a search session is 59% [41]. The average person uses a query length of 1.7 terms [41], such a general query formulation results in a low precision retrieval, yielding many documents that are irrelevant to the information need. That such a large percentage of the retrieved sites, using the most general search engines, are irrelevant to the knowledge gap we are trying to satisfy, shows that even adults have trouble finding the information they are looking for on Internet, let alone children. Due to a smaller vocabulary and less global knowledge of the world around them, children are less capable of formulating their information.

There are various dangers facing children on the Internet, ranging from those that victimize the child, such as being unwillingly exposed to pornographic material; to those that enable children to conduct activities such as downloading patents for bombs, or easily purchasing drugs [44].

Children may unintentionally attain access to irrelevant or unappropriate by typing innocent but imprecise queries resulting in misdirected searches. Many popular search engines can lead an unsuspecting child to numerous porn sites. In an effort to increase traffic to their sites, many web-sites have misleading titles or terms, using popular terms, spelling variations or errors, to abuse the manner in which search engines retrieve documents. An estimated 25% of porn sites misuse popular brand names such as Disney, Barbie, and Nintendo [45] resulting in retrieval of web-sites or documents irrelevant to the user query. To emphasize the scale of the problem we did a short investigation. Using ”Google” approximately one quarter of the sites retrieved for the search query ”Pokemon” contain adult material. Other examples with similar alarming results include words such as toys, boys, Britney Spears and dogs, query terms likely to be typed in by a child not interested in pornographic images [30] [54]. Another example are misleading URLs. For example, `www.whitehouse.com` does not lead to the official site of the White House (`www.whitehouse.gov`) as possibly expected, but rather to a pornographic site.

Furthermore, children are much more likely to click on advertising banners and icons than adults, not being able to distinguish between editorial content and advertising [69]. Research has shown that advertising ranks surprisingly high as a tool to discover new websites [57]. The result is that children are likely to be attracted away from a sites’ editorial content by web advertising and popup windows, and towards unappropriate (including adult content) sites. Furthermore, children are more vulnerable to the commercial promotions and e-commerce, tempting them to actions with serious consequences including making purchases that exceed their pocket money, using up their savings on frivolous purchases, or buying prohibited products or products that are too expensive. Young people often have trouble understanding the real value of money. They may even let themselves be deceived by a promising ad that does not correspond at all to the product delivered.

Children should be capable of independently finding information which adheres to their interests and level of knowledge and development. This is currently not the case; classic tools do not adequately associate with the way in which children think or work. Not only are many of the sites retrieved not relevant to the information need, a rather large percentage contain information not suitable for a general audience. The retrieval of a large number of irrelevant documents is undesired by the users, also parents, teachers, librarians are disturbed by the content of some web-sites retrieved on simple children's queries.

Obviously, when someone intends to do so, many unappropriate documents and web-sites can be freely attained. Using "Google" it took one of us just under a minute to find a recipe for building a Napalm bomb. However, where there is a will there is a way. Protecting children from intentionally trying to access particular information goes beyond the scope of this thesis. This thesis concentrates on a more precise retrieval of relevant documents in relation to the search query and thereby leaving out irrelevant and unintended documents. Whether an additional filtering on particular themes and terms is to be done, to partially restrict intentional unappropriate searches, is left open for discussion with the stakeholders and will become clear during requirement analysis.

The problem as depicted is two-faced. The largest problem is that the many docs retrieved do not satisfy the information need of the user. A side-effect thereof is that, for the sakes of parents, teachers and carers, we must often protect our children from those irrelevant sites that are retrieved.

1.2.2 Objectives

A feasible manner of improving a child's access to relevant documents on the Internet is by offering a search engine suitable to the requirements of children and their guardians, designed specifically for children ages 8 to 12. **The aim of this project is to successfully design and implement an easy to use tool which supports children in their search for information on the Internet.** The fundamental notion is to make searching easy, fun and meaningful in order to improve information accessibility.

The primary focus is on offering documents which better satisfy a child's need, with emphasis on appropriateness of content level. Not only should retrieval result more adequately meet their information need, but irrelevant documents should not be retrieved, and as such, protect children from unwanted and unappropriate web-sites.

The secondary focus is on designing and implementing a user interface that better adheres to the manner in which children think and work. The accent lies on simplification of the query formulation process and navigation, while offering a fun and easy interface to work with.

Furthermore, because we strive for worldwide applicability, the tertiary focus is on language applicability. The program will be made multilingual with target languages being Dutch, English and Spanish for both the user interface and retrieval documents.

1.3 Justification

1.3.1 Social Relevance

World

The large media coverage and amount of attention that the dangers of Internet for children is getting [55] [51] [56] [53], shows the demand for advancements in this area, yet the tools momentarily offered, restrict the informative capabilities of the Internet so dramatically that they are often not used, hinting that the public is still waiting for a tool that better adheres to their requirements.

The age group (ages 8 to 12) has been chosen due to the fact that these children often use Internet to find information (for example, to make a report for school or play games), are familiar with using word based search engines (capable of typing and familiar with using words to formalize needs) but have difficulties formalizing their information need accurately which often results in retrieval of many irrelevant web-sites. Furthermore, this age group is easily influenceable, on a social level, increasing the need to protect them.

IBM

Originating as an American company and now prominent world-wide, IBM has always invested in corporate community relations, paying close attention to their societal responsibilities. While making an active effort to promote human welfare, enhancing relationships with customers and employees, IBM hopes to be seen as a great employer, a trusted corporate citizen and a valued member of the community. Their efforts have shown incredible results, scoring in the top 15 per cent of the 50 most prestigious consulting firms [5], Market Cap's top 100 companies [3], and Business Ethics' 100 Best Corporate Citizens every year since the list's initiation [1], based on social analysis profiles.

With IBM's philanthropic programs, it has become one of the largest corporate contributors to nonprofit organizations and educational institutions across the world. IBM's efforts are focused on helping people use information technology to improve the quality of life; using IBM's expertise in technology and demonstrating their reputation as a solutions provider.

IBM believes that their role in social, human, and legacy issues is not all that different from their role in technical fields or financial markets. "Corporations thrive only in communities that thrive" [6]. Since their main competitive strength lies in technology, they offer programs in which employees help children of low-income families learn the use of technology or advise public schools on information systems [2]. IBM contributes their resources, skills and technical innovations into helping individuals, schools and entire communities thrive.

To give an impression of ways in which IBM is momentarily striving to achieve these goals, only a handful of all the existing programs are mentioned below:

- Reinventing Education:
  Using technology as a leverage for change, it was designed to raise the quality of teaching and learning for all children. [16]

- IBM KidSmart Early Learning Program:
  This program is aimed at giving pre-kindergartens the tools they need to enrich teaching and learning, making children all over the world prepared and excited to learn by increasing their communication skills before they learn to read and write. [17] [15]

- TryScience:
  A worldwide online science and technology center, stimulating interest in science both on-line (www.tryscience.org) and off-line (IBM kiosk in NEMO in Amsterdam [15]) aimed at children ages 8 to 14. TryScience offers a new way for people everywhere to discover the science presented by museums around the world by means of interactive exhibits, adventures, live camera "field trips" and experiments. [19]

- MentorPlace:
  An online mentoring initiative in which IBM employee-volunteers work together with students and teachers to provide students with academic assistance and career counseling, giving them role-models and interpretations of working and working in ICT. [18] [15]

- Webschool:
  An electronic learning environment in the Wilhelmina children's hospital available to children, parents and teachers. It is an initiative to prevent chronically-ill children from complete isolation and falling behind in their education. [15] [20]

As the above list of existing programs shows, IBM is all over the place, actively participating in the education and well-being of children in schools, children's hospitals, family restaurants [52], and children's science centers, only to name a few.

IBM has a strong overall commitment to public education and raising student achievement, promoting the notion of technology access and utilization by all citizens. Using technology as a leverage to make children capable of independently finding information which adheres to their interests and level of knowledge and development clearly fits into IBM's philanthropic efforts. The fundamental notion of improving information accessibility on the Internet by making navigation and search easy, fun, and the results meaningful connects with IBM's efforts in triggering curiosity and stimulating children all over the world to learn. Furthermore, a multilingual solution corresponds strongly to IBM's global approach and philosophy. The goal is thus to develop a tool which encourages children to independently search information on Internet in a safe manner. Such a tool can be which can be advised to children, parents, schools, family restaurants and businesses, children's hospitals and libraries to make the Internet a safe haven for children while providing academic assistance and allowing children to fulfill their curiosity.

1.3.2 Scientific Relevance

Management and Technology

Most software projects fail because the tools devised do not adhere to the users' demands. In the United States, about 80% of software projects are restarted, mostly because they failed to deliver the value expected by their customers [29]. The most important success criteria of successful projects were user involvement (35.1%), requirements management (24.1%) and planning (9.6%) [27].

Before making a specification of a product, one must be sure that it adheres to both the end-user requirements and the requirements of the company funding its production. The product will be brought on the market free of charge with the company's intention of goodwill for society. Decision makers which influence whether or not the product will be used are the user themselves (children), but also parents and teachers, making success of the product (and thus also the company's image) lie heavily on compatibility with these groups.

An interesting aspect is that of the user group. Momentarily research is being done into usability aspects and software requirements of children [47] indicating awareness of the subject. However, extensive models describing how to systematically research and test user aspects related to learning and independently acquiring information is very sparse [73] [65] [70]. Until recently, software testing for this user group has always been done by adults, yet publications are appearing in which the need to include the user in the design process of children's computer products are addressed [66]. We know little about the service requirements that children pose on a search engine, the manner in which they search on the Internet, and the types of information or documents they search for. Determining a method for an adequate user requirements analysis is a difficult task; traditional methods of surveying and invoicing will most probably not yield appropriate results, while other methods such as watching how the users work and the think-aloud-protocol probably will. A model will be devised to research this topic, including a systematic method of defining and describing the problems that each stakeholder involved may encounter and how to come to these conclusions. The method or model used will be evaluated along with its results.

An extensive requirements analysis and marketing research must be completed to model the requirements of all the stakeholders involved in the project. Contributing to a requirements definition are not only the user requirements, but also the parents, teachers and IBM's corporate strategy. Furthermore law enforced requirements such as those posed on children's toys must be researched for applicability.

Who are the different stakeholders involved when making an application to aid children in their search for information on the Internet? Which problems are encountered, to which extent and under which circumstances do they consider it to be a problem? Which requirements do the different stakeholders involved place on a tool to help deal with these problems?

Computer Science

The computer science aspect of this project is two-faced, with specific requirements in both the information retrieval process and the user-interface (formulation input and result output).

User profiling research must be completed to determine the overall characteristics of the types of documents that must be retrieved, and those that must absolutely not be retrieved. Furthermore, a strategy must be devised and implemented to retrieve these documents from the Internet and filter out undesired documents. Needed is to find an appropriate balance between a high precision and high recall. Recall and precision are the fundamental parameters defining the behavior of an information retrieval system. Precision is a measure of accuracy describing the proportion of returned documents being useful, in this case reflecting the probability of filtering out all unwanted documents. Recall describes the proportion of retrieved relevant documents in comparison to all relevant documents on the Internet. A high recall reflects the usefulness of the retrieved documents as described by the users themselves, and is necessary to ensure that the product will be used at all. Unfortunately, precision and recall are inversely proportional.

Many current search engines allow query input by means of a text box, making search initiation difficult when the information need is not clearly defined and ineffective if the user does not understand the search method. Other search engines restrict the user to a selection of predefined categories to choose out of, with the advantage that the hierarchical nature reduces the number of irrelevant documents but are restrictive because they often only contain a relatively small number of documents and the categories can be ambiguous [83]. Challenging is the creation of an alternative graphical user interface which aids in the search through hyperspace, simplifies query formulation and furthermore adequately adheres to the users' characteristics such as age group and cultural backgrounds while remaining intuitive, easy to use and easy to learn.

Yet another challenge is the multilingual aspect. Because retrieval models are language dependent, research must be done towards application of the devised tool for retrieval of documents in other languages.

System requirements must be established and the client made aware of these. Issues such as hosting and minimal system requirements will be determined.

In accordance to the requirements specification, a full working program must be delivered with an adequate user-interface verified against the users' needs. Research must be done into the types of algorithms and techniques which can be used to create the desired product, both behind the scene as on the screen. These do not merely include the graphical user-interface, but also methods for retrieving and filtering documents from the Internet. Furthermore, the application must be able to run online and be responsive, retrieving results real-time.

Which relevant problems can be solved by the implementation of a tool, what are the origins of these problems and how can they be solved? Is it possible to adhere to their needs by increasing retrieval precision, delivering relevant documents and filtering irrelevant documents? What are the specifications of an on-line multilingual application which is intuitive enough for children to work with yet strong enough to aid them in their search for information on the Internet?

1.4 Purpose of Document

On a global level, this document plays two major primary roles. The first is the justification of the conclusions of our academic research, which has mainly been written for our supervisors at the University. It justifies that our research and conclusions are valid, precise, reproducible and integer. Second it describes the structured and systematic process with which we come to our results, which is mainly interesting for our supervisors at both the University and IBM, and any other trainees at IBM, because it justifies and reflects the quality of our conclusions and the product we built by formally describing each translation step from distilling requirements to incorporating them in the final product. Either way, it shows that we are capable of attacking an immensely large and new problem systematically and able to justify and document decisions and steps made along the way, in such a manner that any person following a similar process can use what we have learned to their advantage.

Though this is the primary purpose for which we wrote this document, it does not mean that the document may not be of interest to any other parties. Any person who is working on a similar project, or plans to do so, may be advantaged by reading this document. Our results may be used as intermediate results for related projects.

As such, this document will be given to the project supervisors and also placed in the IBM library for traineeships, as a form of knowledge management. Copies will be given to the faculty exam commission as justification of our work and the evaluation of our supervisors. It will furthermore be given to some friends and family.

1.5 How to Read This Document

This document has been built incrementally and iteratively. It describes the process of the project while it was in progress. As such it describes what has been done at what point in time and who was involved. It describes problems encountered and accomplishments achieved.

To give the reader a clear description of the process of the project we have chosen to collocate the chapters as a diary, each section describing a different subgoal, plan and milestone, yet all leading towards the same overall goal, the accomplishment of the project as a whole, which has and could only be achieved by defining and accomplishing each subgoal in the process. As such, the document has been divided into chapters describing the time-line and process of achieving each of the milestones.

- *Project Plan:* Chapter 2 identifies the intention and objectives of the investigation, as well as stating the research questions to be answered. It also provides a general product description of the prototype to be built and identifies the activities and to be performed and the research to be completed in order to create it. Furthermore it establishes schedule baselines and provides a basis for tracking and control.
- *Requirements Analysis:* Chapter 3 describes both the approach, execution, and results to requirements distillation for the specification of the product to be built.
- *Design:* Chapter 4 describes a plan of how the process of implementing Giggle was to take place. The specifications of the Requirements Analysis document have been translated into formal models which form the basis for the implementation phase and describe a breakdown of the system into multiple components which can each be coded and tested separately.
- *IR Research:* Chapter 5 describes the research we have completed with respect to aspects pertaining to Information Retrieval. It describes hypotheses and goals, including the results of the investigation with respect to usefulness for the product and contribution to Information Retrieval research on an academic level.
- *Implementation:* Chapter 6 describes the execution of the plan introduced in the design chapter (4). It describes which decisions were made during the coding and programming stage of the project and the capabilities and restrictions of the resulting programs.
- *Testing:* Chapter 7 describes the how the functions and programs built during the coding stage of the project were tested according to the design guidelines described in chapter 3 and the design models in chapter 4. This chapter also includes a description of the system (hard and software) requirements posed by the product.
- *Acceptance and Finalization:* Chapter 8 describes the steps taken to turn over the product to the client to ensure that it can be used in the future. This section includes a reflection on the business requirements posed on the product, and an evaluation of the tool and process approach, including limitations that have effected the research results. It also describes which aspects of the project can further be elaborated on in the future.
- *Discussion:* Chapter 9 describes the final and overall results of the projects, both in terms of goals accomplished and those not accomplished. It also includes a personal reflection by each of the project owners, indicating their contributions.

For clarity and consistant interpretation of terms, we refer to term definitions in the glossary at the end of this document.

Chapter 2

Project Plan

2.1 Executive Summary

This chapter, the Project Plan, is useful for any person directly involved in this project. Its content is imperative to ensure project progress and quality of process approach. It establishes schedule baselines and provides a basis for tracking and control, and furthermore describes the roles and responsibilities of all people involved in the project. It describes the global approach to the project and systematically breaks the process down into manageable phases (each testable on quality and progress), including a description of activities, milestones and deliverables of each phase and the project as a whole.

2.2 Introduction

The previous chapter gave an introduction to the "Giggle: Search Engine for Kids" project, including a global description and justification for its existence. The intent of this chapter is to provide information about the process, deliverables and a schedule of the project as a whole.

2.2.1 Goal of Project Plan

The goal of the project plan is to determine and establish a plan to achieve and break down project goals, in order to successfully design and implement an easy to use tool that supports children in their search for information on the Internet.

The main project goal has been defined in section 1.2.2. In order to maintain overview, it is imperative to establish a plan with which this goal can be achieved. A systematic approach is necessary to guarantee both quality and progress of the project, and to ensure that all goals are met. This plan will provide a general product description and identifies the activities to be performed in order to create it.

Though this chapter is not a mandatory aspect of an academic thesis to justify research conclusions, it establishes schedule baselines and provides a basis for tracking and control, imperative to ensure any goal what so ever will be achieved. It shows that the authors have thought out what they plan to achieve, and systematically broken that down into manageable steps which, when executed serially, will result in both the desired research and product goals.

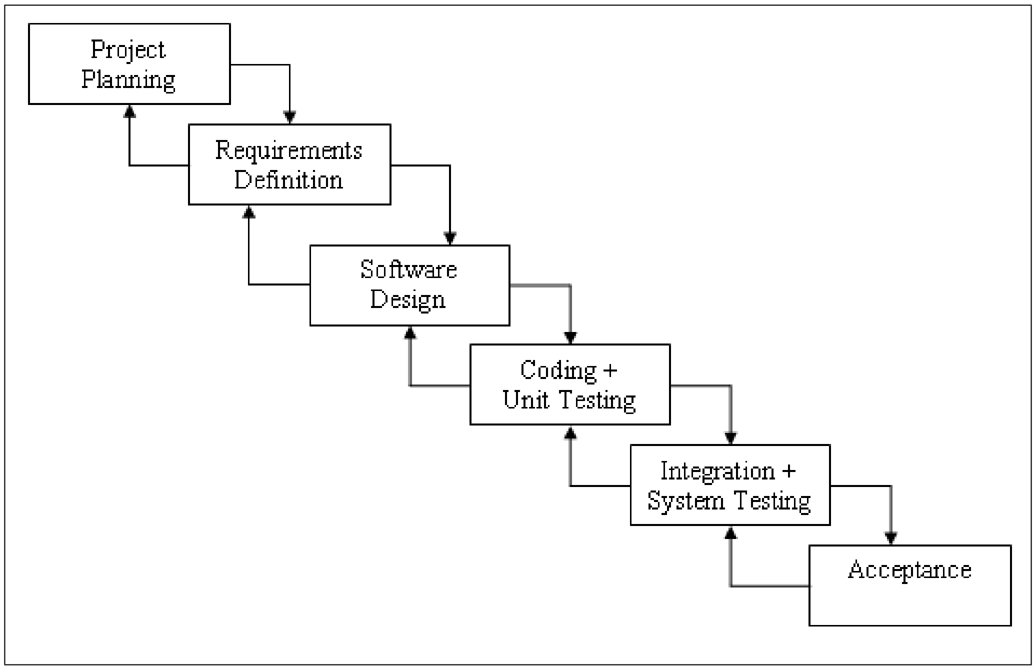


Figure 2.1: The Waterfall Model

This plan has been written as a primary form of communication, to ensure that all people directly involved in the project (thus all supervisors and team members) know what to expect from each other, when to expect it, what not to expect, and what is expected from them. Such a plan, together with a process evaluation (See section 8.5.2) can also be interesting for any other person wanting to set up a similar type of project.

2.3 Project Activities and Products

2.3.1 Process

Doing good research means collecting and analyzing data in a structured manner and working towards a well-defined goal. Making an appropriate software tool to solve a problem can only be done if the software delivers the required functionality and performance to the user while also addressing issues of usability ([26]). This section gives an overview of the process with which we plan to achieve good research and a good software tool. Because of both its strength and simplicity, we base our software development on the waterfall model (see figure 2.1). Though the waterfall model as a whole is an incremental process, phases within the model can be either approached incrementally or modular, depending on the project constraints and the goal you wish to attain. Modularity is chosen when one wants to ensure all requirements are taken into account. Incremental is chosen when, often due to certain constraints, when one wishes to ensure the most important requirements have been taken into account. The different process phases and the approach generally taken for such a project are summarized below ([24]):

- **Project Planning:** Modular approach to scheduling baselines and establishing people's roles and responsibilities.

- **Requirements Definition:** Modular approach to establishing the system's services, constraints and goals by consultation with system users.

- **Software Design:** Modular approach to defining the requirements in detail and translating them into a formal design model which serves as a system specification.

- **Coding and Unit Testing:** Incremental approach to realizing the software design as a set of program units. Unit testing involves verifying that each unit meets its specification.

- **Integration and System Testing:** Incremental approach to integrating and testing the program units together as a complete system to ensure the software requirements have been met.

- **Acceptance:** Modular approach to delivering the software system to the customer.

Only after completion of each phase does the development go onto the following phase. See section 2.3.3 for more details about each phase. During cascading, the stages feed information to each other, the output of one stage being the input of the next ensuring a process flow. As such, if problems in requirements are identified during design, the model allows a step back to fixing the problem. Thus, this software proces is not a simple linear model but involves a sequence of iterations of the development activities.

The preliminary research includes investigating and analyzing the issues corresponding to the information need of children and possible solutions for dealing with this need (see section 1.3.2). In order to distill all the information required during the implementation process, this process is roughly broken down into two aspects, the child-related aspect and the tool-related aspect. The first involves research into the user group and other stakeholders involved. This includes researching cognitive characteristics, capabilities, knowledge levels, and information needs of children in order to create a specification that adheres to the needs of all stakeholders. The second aspect involves investigating the existing state-of-the-art products and solutions for these types of applications, in order to model modern capabilities, in products and research, and the problems and issues associated with them.

To deliver the required functionality and performance to the user, a user requirements analysis will be completed. This is necessary to determine the product requirements and to ensure the quality of the product. In the initial stage a plan will be drawn out, including which questions are to be answered about the users (and other stakeholders) and their environment, how to find an approach to answering these questions, and how to analyze the results and thereby distilling the requirements of all stakeholders. Investigation will also be done into different manners of distilling these requirements. The requirements will partially be based on the results of the preliminary research, and partially based on own experiments and interviews with users and stakeholders (parents, teachers, and customer).

Subsequently we will translate the user requirements into a design. The resulting design will describe functional and non functional requirements, and also system and hardware requirements for the tool.

The design will incrementally be translated leading to the implementation of a working program. Implementation and testing will be done incrementally and repeatedly interleave

Project phase	Deliverable
1. Orientation	Project Proposal, Project Plan
2. Preliminary Research	Literary Research
3. Research & Definition	Requirements Analysis
4. Design	Architectural design
5. Implementation	Working prototype
6. Testing	Tested program
7. Acceptance & Finalization	Final product, Thesis, Presentation

Table 2.1: Project phases and their deliverables

each other, continuously verifying if the program's behavior corresponds to the user and design specification.

After verification, the program will be turned over for acceptance and embedded in an environment for use in the real world.

2.3.2 Project Schedule

The project has been broken down into 7 subsequent phases, each constituting its own activities, milestones and deliverables with each phase (see section 2.3.3 for more details). Table 2.1 shows which stages of development can be distinguished and the resulting deliverable expected from each. The thesis project comprises 21 work weeks, however will be spread out over a longer period of time. The thesis will be written throughout the entire project. A long-term planning that has been made to give an impression about the duration of each phase and when each of the deliverables can be expected (See Figure 2.2).

2.3.3 Project Phases

The proces has been set up as a simple waterfall model. Every phase is followed by a subsequent phase with its own deliverable constituting the results of the milestones.

This section gives a more extensive description of the phases and their deliverables.

Orientation

This is the initialization phase of the project in which a Project Proposal and Project Plan will be produced. The main deliverable, the project plan (this document), can be seen as a manual for the entire project.

Preliminary Research

During Preliminary Research phase a literary research will be completed, including investigation into user group characteristics and existing technologies. The following aspects will be investigated and analyzed:

- Literary research and interviews with IBM employees to investigate IBM's corporate strategy, existing projects, and their criteria on the product and process.
- Stakeholder and user group research:

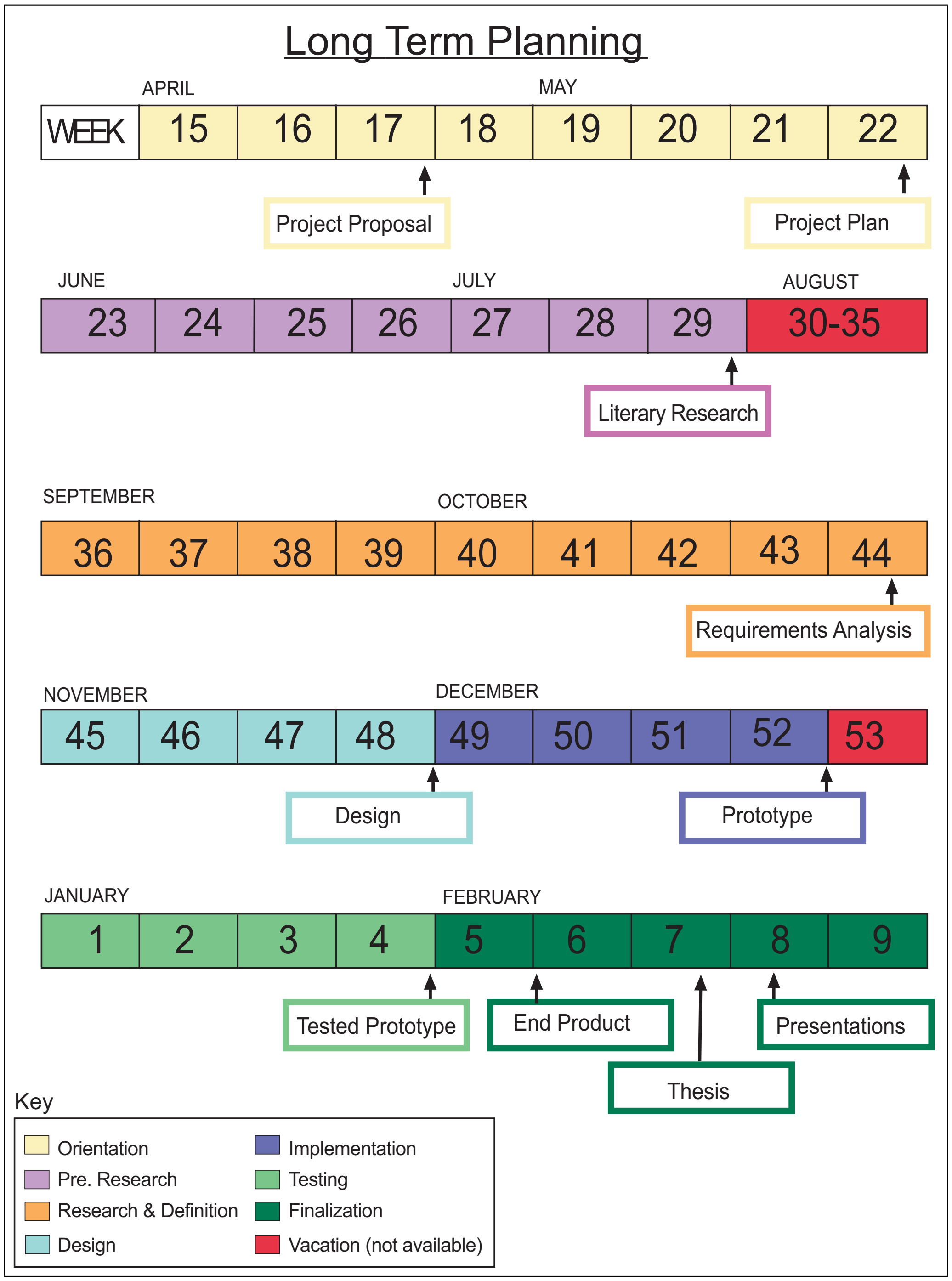


Figure 2.2: Long-Term Planning

  - Interviews to distinguish between different stakeholder groups and their roles.
  - Literary research and interviews with experts to devise an appropriate plan and method for acquiring stakeholder and user requirements.
  - Literary research into previous research on the user group.
  - Literary research, interviews with stakeholders, and viewing user group (using methods such as the think aloud protocol ([64]) and post task interview) at work to complete user group requirements analysis. Subgoals are determining cognitive characteristics of children, see how they think and work using existing tools and systems, determine deficits of current tools and possible improvements, yielding a sketch of user group's knowledge about existing systems, verbal competence, and extroversion levels.
  - Interviews with stakeholders and users must yield information and language knowledge level, and determination of types of documents of interest, yielding information for parental and filtering requirements analysis, their standards and expectations.

- Investigation into existing means, technology, tools and solutions and the problems that are encountered with respect to requirements related to the same user group.

This activity will result in a report describing the results of the preliminary literary research.

Research and Definition

During this activity an definition study analysis will be completed. All stakeholder requirements must be distilled during this phase, resulting in the product requirements. Here, the boundaries on the project will be defined more precisely. Where necessary, an appropriate compromise between conflicting requirements must be made. The literary research (see section 2.3.3) will be used as a foundation for this activity.

This activity will result in a Requirements Analysis document describing the product requirements and defining the boundaries of the project.

Design

The goal of this activity is to create a complete architectural design of the final product according to the requirements described in the Requirements Analysis document (see section 2.3.3).

Literary research and user group viewing will be done to determine effectiveness of different types of user interfaces, and visual representations of document content. Furthermore, literary research into effective information retrieval and filtering algorithms and multi-linguistic information retrieval will be completed as also the possible technical capabilities available to answer stakeholder requirements. Furthermore, research will be done into existing state-of-the-art tools and programming languages which could be used to create the desired application.

The design activities will result in an architectural design including an interface design and system requirements. The design plan will describe and divide the subsequent activities into several disjunct parts in order to maintain overview during the implementation phase. The system and other technical requirements will be discussed with the client.

This activity will result in a Design Plan describing the final product, how this is to be made, and the considerations made along the way.

Implementation

A complete and working prototype will be built during this phase. The product will be built according to the Design Plan described in section 4. After the prototype has been built, it will be shown to the client for approval. Furthermore, the stakeholder groups will be asked to give an impression of the program and any ideas, suggestions or improvements.

Testing & Validation

When the implementation has been completed, thorough testing will take place to see if the product adheres to the specifications. The program will be tested by us, several user group members and involved stakeholders. The functionalities of the product will also be tested and verified, to ensure that it does what the customer and users expect and want. Faults or deviations to the specifications will be reported in the product specification. Product faults must be recovered and fixed in order to guarantee an adequately working product.

Acceptance & Finalization

During the Acceptance & Finalization phase the product will be turned over to the client for acceptance. Steps for acceptance will be devised. Together with the client the steps in the acceptance plan will be completed to make sure that the product is embedded and can be fully used, building awareness for the tool among all stakeholders involved – users, parents, teachers and IBM.

Furthermore, during this stage the thesis will be finalized and presentations will be held at the KUN and IBM describing the project. The thesis describes both the method and the results of the requirements research as also the computer science techniques used during development of the application. Furthermore, a final evaluation of both the tool and process approach will be described in the thesis.

2.4 Project organization

This chapter gives general information about the project and those people involved.

2.4.1 Project Tracking, Monitoring and Control

To monitor progress, milestones will be tracked against what was planned. If problems are encountered in the development of the project, they will be reported to the supervisors. Meetings with each supervisor will be set up on a regular basis to monitor the overall progress of the project.

2.4.2 Involvement, Roles and Responsibilities

The following defines the roles and responsibilities of the members involved in the support and approvement of the project and its development:

- Renske Weeda: project owner
  Responsible for setting up, executing and completing the project, and also guarding progress and quality of the project as a whole. Will be involved in each phase of execution.

  Primarily responsible for process (devising a plan for) and progress of execution of the orientation, preliminary research, research and definition (requirements analysis), and product design phases. During design phase, focus will be on interface aspects of program.

- Jolanda van Gendt: project owner
  Responsible for setting up, executing and completing the project, and also controlling progress and quality of the project as a whole. Will be involved in each phase of execution.

  Primarily responsible for process (devising a plan for) and progress of execution of implementation, testing and acceptation phases, with emphasis on quality control. During design phase, focus will be on technical aspects of program.

- Franc Grootjen: supervisor KUN (computer science)
  Provides overall support and guidance throughout the project, with emphasis on progress control. Supervises daily planning, progress will be monitored in weekly meetings problems encountered and milestones achieved will be discussed. Strong involvement throughout the entire project, with more emphasis during the second half in which the computer science aspects of information retrieval, design, implementation and testing play a larger role.

- Pepijn Vos: supervisor KUN (management and technology)
  Provides support and guidance on management and technology aspects throughout the project. Involvement throughout the entire project, however more emphasis lies on the first half and finalization stages of the project during which requirements analysis and acceptance play a dominant role in his field of expertise.

- Warner Dijkhuizen: supervisor IBM:
  Acts as the client's single point of contact for reviewing and approving project deliverables and changes. Evaluates the application for overall operability at acceptance. Responsible for aiding in the acceptance of the product, aspects of embedding the product and increasing awareness of its existence. Guidance is expected within the organizational structure of IBM, and also introduction to contacts (both within IBM and with similar projects) for interviews, experiments and testing of stakeholders.

2.4.3 Contact Information

Name:	Renske Weeda
Company:	KUN
Function:	Student
Tel nr:	+31 (0) 63 028 0136
	+31 (0) 24 36 52 165
Email:	rweeda@sci.kun.nl
Relation:	Project Owner

Name:	Jolanda van Gendt
Company:	KUN
Function:	Student
Tel nr:	+31 (0) 62 343 0274
	+31 (0) 24 36 52 165
Email:	jolandag@sci.kun.nl
Relation:	Project Owner

Name:	Warner Dijkhuizen
Company:	IBM NL
Function:	Corporate Community Relations Manager
Tel nr:	+31 (0) 65 324 1708
Email:	warner_dijkhuizen@nl.ibm.com
Relation:	Supervisor, client

Name:	Franc Grootjen
Company:	KUN
Function:	University Lecturer, Computer science faculty
Tel nr:	+31 (0) 24 365 2283
Email:	sparky@cs.kun.nl
Relation:	Supervisor, Computer Science

Name:	Pepijn Vos
Company:	KUN
Function:	University Lecturer, Management faculty
Tel nr:	+31 (0) 24 361 1515
Email:	P.Vos@nsm.kun.nl
Relation:	Supervisor, Management and Technology

2.5 Final Result

2.5.1 Deliverables

The following final products will be yielded as a result of the project:

- Requirements Analysis (including literary research)
- Design Plan

- Final Application
- Thesis
- Presentations at the KUN and IBM
- Evaluation and reflection

2.5.2 Evaluation

Final evaluation will be done according to the deliverables mentioned above. Because this is an academic master's thesis, the grade will primarily be determined by the academic supervisors at the KUN. The computer science aspect of the project comprises about two-thirds of the whole, while the management and technology aspect comprises about one-third of the whole. The final grade will be determined in accordance to these weights. The supervisor at IBM will also deliver a grade, capable of rounding the final grade up or down by a maximum of one point.

2.6 Conclusion

The intent of this chapter was to provide information about the project process, in terms of a deliverables, schedules and responsibilities. The project has been broken down into several phases, each having its own activities, milestones, objectives and deliverables. The output of each phase is the input of the next phase. As such, the the project plan ensures that from start to finish, the primary goal established is to be achieved by systematically completing smaller, more manageable steps which can each be evaluated on aspects of quality and progress.

Chapter 3

Requirements Analysis

3.1 Executive Summary

In order to assure that the project goals are achieved, by definition of our goal, it is imperative that we correctly determine what the requirements of our focus group and all other stakeholders are. This chapter describes both the approach, execution, and results to requirements distillation for the Giggle Search Engine for Kids product. Observation and interviews of children, interviews with IBM employees and people involved with similar projects, interviews with carers and educators, and literary research are described in section 3.3.3 and has yielded an extensive list of stakeholder requirements. Conflicting requirements have been resolved by manner of compromise, resulting in a full list of requirements which form the basis for design (section 3.4.2). Furthermore, a prioritization for implementation has been developed (section 3.5.1) justification of which requirements will not be implemented (section 3.5.1) have been described (including a list of features to be implemented in the future during further development of Giggle).

3.2 Introduction

The Project Plan introduced some of the problems at hand with Internet search engines currently available. Research has shown that primary school students need assistance if they are to find information successfully, and that such assistance must be provided either by teachers or by the Internet portals themselves ([76]). When children use search engines in their search for information on the Internet, a lot of the results retrieved have nothing to do with their information need. This has shown to be problematic, not only for the user who has to click through the resulting list in search of documents that are truly relevant, but especially those web sites that are not relevant often contain content that is inappropriate for young children. Many people involved in the upbringing and education of children (such as parents, teachers and even politicians) are not happy with this evolvement because it exposes children to aspects (such as pornography, buying of drugs and other criminal activities) they try so hard to protect their children from in everyday life.

This chapter describes the approach to be taken in distilling and analyzing all the requirements for the software tool to be built, as described in the project plan. The requirements found are to be translated into an appropriate design, which will in turn be implemented, yielding the resulting software tool.

Sections 2 and 3 focus on the approach to be used for requirements distillation. Section 2 describes our information questions, which will help us to systematically compose an accurate and complete list of requirements. Section 3 describes our research strategy, how we planned to get the answers to the questions posed in section 2 (determining who the stakeholders are and which methods we can use to distill the requirements of each stakeholder group). In section 4 we describe the results of the distillation processes, the requirements we found, and the product characteristics that are or are not to be translated into a design. We end this document with a conclusion, which explains the restrictions on our investigation, prioritization of the accepted requirements, and design guidelines. As implementation is an iterative process, prioritization sets up a planning scheme just in case time becomes a constraint; in that case it will not be possible to fulfill all the requirements, but our approach does give assurance that the most important characteristics will be implemented, as such the resulting product may not be the best product buildable, yet still be capable of primary purpose, being easy to use and supporting children in their search for information on the Internet.

This document has primarily been written for the contract owners (us) to clearly establish, identify, and document product requirements. However, it will also be used between us and the client (IBM) as a manner of communication, to establish correct expectations, and ensure agreement, about what characteristics the product to be built will and will not have.

3.2.1 Goal of Requirements Analysis

The goal of the requirements analysis is to determine the relevant requirements posed, in order to successfully design and implement an easy to use tool that supports children in their search for information on the Internet.

Relevant requirements are defined as those that exhibit a relationship to an easy to use and supporting tool for children while searching for information on the Internet. The translation of the design into an operational tool, the final goal of the project, is achieved by implementing the characteristics defined by business requirements, functional requirements and non-functional requirements. In order to make this possible, the requirements posed by all stakeholders are to be determined and defined in the course of this engineering phase, the requirements analysis.

Subgoals

To complete the requirements analysis in a structured manner, the goal has been divided into smaller, more manageable questions which, when answered, will fulfill our information need, making us capable of understanding the aspects required in the tool's design. Upon answering all of these questions we ensure ourselves to have gained sufficient knowledge on the topic to answer the research questions, and until then, the requirements analysis has not yet been completed.

The goal of the requirements analysis is to determine the (relevant) *requirements*(1) posed, to *successfully design and implement*(2) an *easy to use tool that supports children on the Internet*(3) in their *search for information*(4). To ensure a good design that adheres to these characteristics, we must research and distill information on these topics during this phase (the requirements analysis), so that subsequently they can correctly be translated into the design of the product. The main questions which has to be answered in order to gain knowledge about these characteristics are:

1. What are the requirements?
2. How can we successfully design and implement our tool?
3. What is an easy to use tool capable of supporting children?
4. What is information search?

The requirements investigation commences by seeking answers to questions 2, 3, and 4, which will implicitly yield the answer to question 1. Specific details about the distillation process can be found in section 3.3.1. These answers will give a sufficient basis to answer question 1. The result is a prioritized list of stakeholder requirements is established (to be found in section 3.4.2), which is furthermore the final goal of the requirements analysis phase.

3.2.2 Relevance

The importance of a requirements analysis has been long recognized. Various studies have shown that errors in the early stages of software systems development (i.e. establishment of the requirements) were the hardest to repair in a later stage, often leading to cost overruns, missed schedules, and low quality software ([72] [82]), and in some cases even leading to undelivered systems. In contrast, proper requirements analysis at an early stage in the development process is argued to yield more successful systems.

The remainder of this section is devoted to describing the relevance of each of the four sub-questions posed above with respect to the requirements analysis, and in essence, to the project as a whole.

Requirements

Requirements are the criteria posed by all the stakeholders. Such a stakeholder is a person or company who has an interest or share in an undertaking, and is in some manner involved in the process or final product. The stakeholders can be said to be the most important people in this project. They determine whether the product will be accepted and ever be used. Meeting their satisfaction is mandatory to ensure the value of this project. To determine what their desires and constraints pertaining to the tool are, we must know 1) who the stakeholders are, and 2) determine how each of them is involved in the project. Subsequently the conditions or capabilities needed by children to solve a problem or achieve an objective (in relation to search of information on the Internet) can be determined, allowing us to comprehend their problems and sources of distress are. The provided knowledge can be used to determine which conditions or capabilities are required.

Successful Design and Implementation

Design and implementation must be successful to ensure desirable outcomes of the final product in terms of quality and functional and non-functional characteristics. A successful design can be described as a sketch delineating main features to be executed to come to a desirable outcome. Successful implementation is the actual accomplishment of translating a design into the desirable physical piece of software. In the end, IBM determines whether the project has been a success. Therefor, it is imperative that desirable outcomes (or aspects which make an outcome desirable), valuable to IBM, must be determined. Furthermore, following IBM's best practice methods, based on their long history and experience in software development (naturally including both successful and less-successful projects), will increase the chance of having a successful design and implementation from a good requirements analysis.

Easy to Use and Support

The Internet is a global network of computers comprised of an unstructured and unregulated data collection. Due to its nature, seeking for information on the Internet can often lead to the retrieval of inappropriate data. Support can be defined as lending assistance to, and aiding when one experiences difficulties or failing in an attempt to achieve a goal. In essence, in order to be supportive to children, it is imperative that the tool is easy to use. An "easy to use tool" is defined as a relatively simple instrument used to perform a task. A tool not pertaining to this characteristic will simply not be consulted, making it useless to the children, IBM and us; its importance must therefor not be underestimated.

Search for Information

Understanding children's search strategies and the types of information they wish to encounter is the first mandatory step in making a solution that adheres to their needs. An understanding must be built about what information (rather than just data) is to a child, and what it means to search this information.

3.3 Research Strategy

In order to systematically attain a complete view of all the information pertaining to the goal of the requirements analysis, we have broken this goal down into several smaller research questions (see section 3.2.1). Choices have been made about which sources to consult and which methods to use for the distillation of information. An overview of relevant distillation methods, methods for the collecting and analyzing data, are described in [22]. Section 3.3.1 describes the process with which we handled the requirements analysis phase as a whole. Section 3.3.2 describes the methods used to distill the required information. Section 3.3.3 describes the different sources we have access to, the quality criteria they adhere to, and the measures taken to ensure the quality of information. Details of execution are described in appendices A.3, and A.4.6.

3.3.1 Research Process

This section describes, globally, the path taken with which we collected and analyzed all the required information and from that came to our final conclusion, the result of the Requirements Analysis. This process can be seen as 4 distinct phases whose course and interaction points can be seen in figure 3.1 and is described below.

Phase I: To start, one by one we analyzed the sub-questions posed in section 3.2.1 and determined which sources available to us could in any way help us in search of our answers. The relevance of each source in line with each specific question is discussed and justified in section 3.3.3.

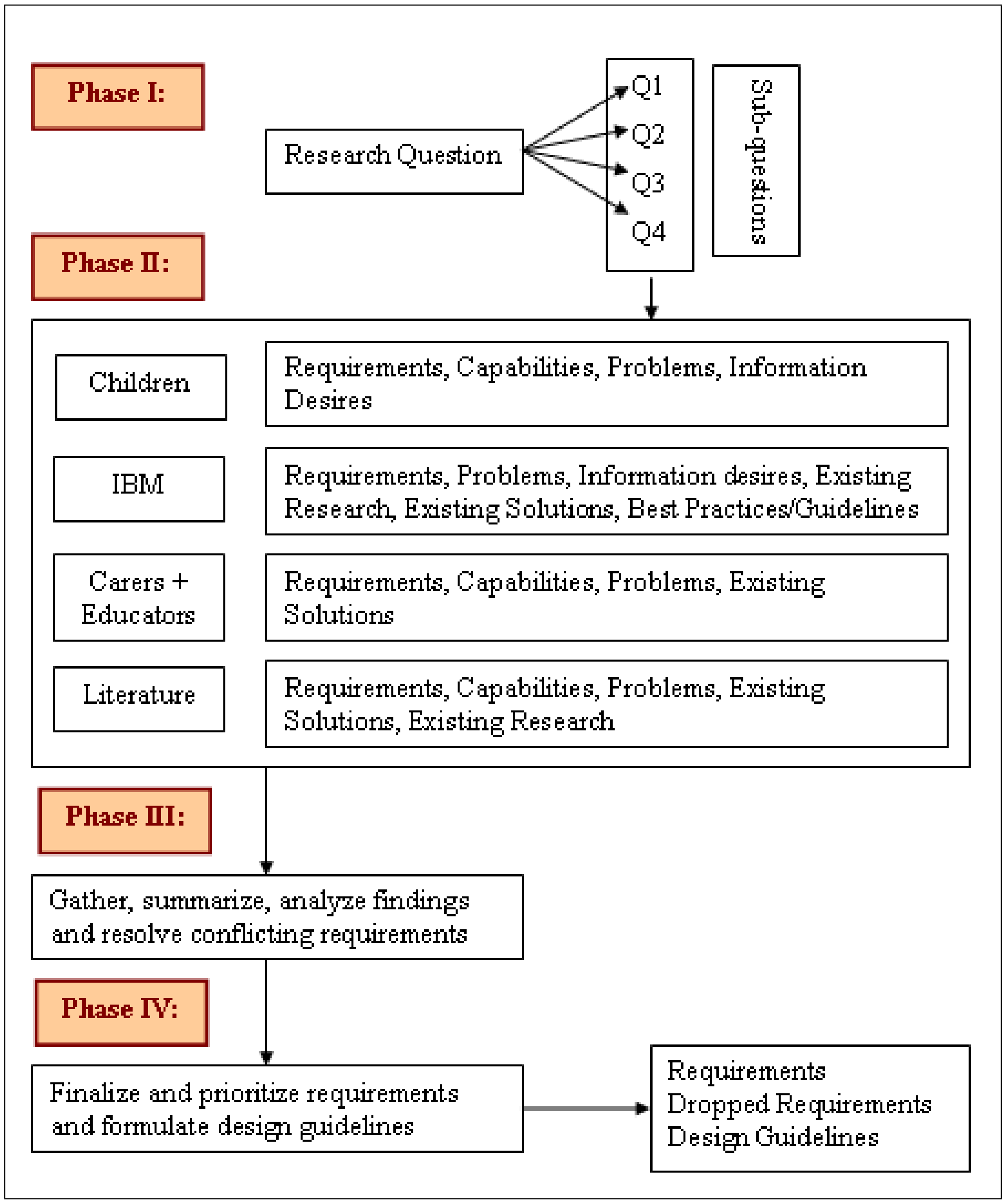

Phase I:
Research Question
Q1
Q2
Q3
Q4
Sub-questions
Phase II:
Children
Requirements, Capabilities, Problems, Information Desires
IBM
Requirements, Problems, Information desires, Existing Research, Existing Solutions, Best Practices/Guidelines
Carers + Educators
Requirements, Capabilities, Problems, Existing Solutions
Literature
Requirements, Capabilities, Problems, Existing Solutions, Existing Research
Phase III:
Gather, summarize, analyze findings and resolve conflicting requirements
Phase IV:
Finalize and prioritize requirements and formulate design guidelines
Requirements
Dropped Requirements
Design Guidelines

Figure 3.1: Description of research process.

Phase II: Subsequently we researched different distillation methods and determined which method would be a best-fit between each question and each source. Table 3.1 summarizes, per sub-question, which available sources can be examined in search of answers and which distillation method is most appropriate for attaining our goal; justification of these choices can be found in section 3.4.1. Furthermore specific aspects pertaining to each interview can be found in the respective appendix pertaining to the source. Because interviewing our sources per sub-question would both be logistically inefficient and furthermore force us to be a nuisance to our sources, we gathered all questions and sorted them by source rather than by sub-question. With these in hand we interviewed our sources one by one (distilling information pertaining to all sub-questions relevant to the source) and analyzed the results separately, allowing us to determine the quality of the data collected and draw conclusions.

Phase III: Subsequently, the sets of results per source were analyzed together, common aspects and characteristics were gathered and summarized. Contradictions and differences were analyzed and justified, discussed in the project group and in some cases even lead to a new appointment with each the contradicting sources for additional feedback. The result was a list of viable requirements, a common denominator of all requirements collected.

Phase IV: Finally, in consent with IBM, a prioritization was made in the requirements. This prioritization can be used to guide the (time and tasks) planning process for the subsequent engineering phase, the design phase.

3.3.2 Data Collection Method

In order to collect information about a problem, decisions must be made about which types of information must be gathered, which structured methods of collecting and analyzing data are to be used, and which particular sources to consult for what type of information. Which distillation method is appropriate is not merely determined by the type of data one wants to acquire but also which data source it corresponds to. Furthermore, we pose quality criteria on the data collected. Which criteria we pose and how these adhere to the sources in relation to the distillation method used is described in section 3.3.3. The remainder of this section describes the above process in relation to each sub-question to be answered to achieve our project goals. Table 3.1 summarizes the result of this process.

Requirements

In order to determine which stakeholders are involved, what their sources of distress are, and what the required conditions or capabilities are to solve this, several sources can be inquired such as children, IBM, carers and educators, and literature (including statistics about children's primary Internet locations, indicating roles of involvement and responsibility).

In order to compose a single prioritized list of requirements, conflicting interests must be resolved. Resolving these conflicts is done by sitting around a table with the involved stakeholders and finding a compromise which is justifiable and each stakeholder group can live by. Discussions with people involved with (or documentation about) similar projects, our client, carers and educators, and children will make us capable of finding such a compromise.

Successful Design and Implementation

In order to determine desirable outcomes in terms of quality and characteristics, and how to assure the fulfillment of requirement conditions in the final product, we can use the organi-

Questions	Sources	Method
Requirements	Children	Interview and Observation of Children
	IBM	Interview Warner
	Carers & educators	Interview parents, teachers, school head, librarian
	Literature	Secondary data (documentary & multiple source)
Successful design & implementation	IBM	Interview programmers and designers, Blue pages
Easy to use and provide support	Children	Interview and Observation of Children
	IBM	Interview Warner and his contacts
	Carers & educators	Interview parents, teachers, school head, librarian
	Literature	Secondary data (documentary & multiple source)
Search for information	Children	Interview and Observation of Children
	IBM	Interview Warner and his contacts
	Carers & educators	Interview parents, teachers, school head, librarian
	Literature	Secondary data (documentary & multiple source)

Table 3.1: Overview of questions to be answered, including distillation source and method.

zation's guidelines, standards, and procedures, and interview IBM employees or people with experience with similar projects. Software designers know how to make a good design, and what pitfalls there are in making a design. Programmers can help us make the right choice of programming language, giving us criteria for successful implementation, and give us tips about how to test the tool at the end of the implementation. Our client contact determines the minimal functionality and other criteria's for a successful tool, and is furthermore experienced with similar projects. IBM's w3 intranet describe methodologies (such as the Quality Management System's Software Design Checklist) to appropriate requirements translation, but interviewing people on the work floor can give us an idea of how holey the methodology is, and by doing so, help us understand the pitfalls in this route, others experiences, and tips and tricks to getting things done efficiently and effectively. The result is to lead design guidelines (see section 3.5.2).

The interviews held mostly contain open questions, allowing the interviewee to define and describe a situation or event, including foundations for their attitudes and opinions. Furthermore, this method allows for a flexible interview (of order and logic of questioning) and allows for an environment where the interviewee is under the least amount of pressure. Because it is of significant value to establish personal contact, these interviews will be conducted face-to-face, on a one-to-one basis.

Easy to Use and Support

In order to fulfil the characteristic of ease of use, we must know which characteristics make an instrument simple for a child to work with and how we can simplify a child's task. Adequate support can only be achieved by understanding which navigation methods can be used to traverse the unstructured Internet and what difficulties and dangers children experience (what barriers they have) while using the Internet. With this knowledge we can find out how, and with which tasks we can lend assistance, which barriers we can alleviate, and how to protect the children from inappropriate data.

Criteria pertaining to ease of use and support are primarily determined by the users. However, as we will see in the section 3.3.3, distilling relevant information from children can be problematic, thus other sources of information should be consulted. In order to locate the barriers that children experience, several sources must be consulted. Due to their involvement and knowledge with respect to computer tools built specifically for children, IBM or people involved with similar projects are to be consulted. Furthermore, because carers and educators are closely involved with children, often offering them assistance during their search, and observing their behavior while on a computer, they may also have a rather good and representative idea of the barriers children face due to the Internet's unstructured nature. Stakeholder groups such as carers, educators, IBM, and those otherwise socially involved may find the unregulated data dangerous for children, and thereby be an important information source describing the problems posed to children on Internet. Academic research, for example documented experiments with children pertaining to this topic, describe additional barriers and cognitive aspects relevant to usability, yet also hint towards possible solutions including navigation methods and search strategies, and can be used to complement and enhance trust in the quality of the requirements found. For further details about the methods, number of interviewees and results of observations and interviews with IBM employees, children, and carers and educators, we refer to Appendices A.2, A.3 and A.4 respectively.

Search for Information

Obviously, because the children execute the search, they determine what they search, why they search, what their personal interests are, what types of results they expect, search strategies, and appropriateness of documents (levels of understanding, adult content). Details can be investigated by participant observation and interviews with children. However, their search task need not always be triggered out of own initiative, but sometimes rather in an educational atmosphere, justifying the need to interview carers and educators (i.e. teachers, school head, and librarians). In order to find more in depth information on search strategies, literature can be used. Reading and understanding levels of children have been researched and algorithms to determine content appropriate to literacy level, have been derived, all documented in literature. Some organizations have published blacklists and content concepts inappropriate to children, which is based on extensive research and forms a reliable source.

3.3.3 Sources and Quality

The information we distill from our sources must satisfy to a number of quality criteria. These criteria are:

- Validity: the information must be sound, with the possibility of proving falseness;
- Public domain: the knowledge and information should be available to others;
- Precision: the information must exactly defined rather than vague;
- Reproducibility: repeating the same research must lead to the same results (information);
- Objectivity: the information must be independent of individual thought;
- Integrity: the information must be sound, complete and incorruptible.

All sources used partly or completely satisfy these quality criteria. Where only partial adherence to quality is achieved, additional sources are consulted to verify conclusion. The following subsections give some general comments about the quality of each consulted source.

When collecting information through observation and interviews, information becomes saturated at a particular point. When no dramatically new information arises during additional observations or interviews, we can conclude the investigation and analyze the results.

Children

Children, being the end-users of the system, are probably the most important source of information because the system is being built for them. We may think that no one knows someone like they know themselves, this is true to a certain extent, but may not completely apply to children. Due to their inexperience with the world around them, lack of self-expression and self-reflection, their opinion may not be objective, integer, nor a fair view of reality. We therefor must not only interview them and listen to what they say, but especially watch how they work in order to distill all the required information through participant observations. Of course it is impossible to observe all the children, so we have to take a representative sample of it in order to satisfy the criteria of the distilled information. Our sample is composed of about ten children of ages 8 to 12 from an international school in this region. We choose for an international school due to the children's multi-cultural backgrounds. Observations made during other (relevant) researches, and interviews with carers and educators can be used to verify our results. This will increase the certainty about the objectivity and integrity of our information. Besides merely interviewing children, usability testing can be executed to determine quality, task performance, and achievement measurements, possibly in comparison to traditional tools. More information about the observation of children can be found in appendix A.4.

IBM

As our client and extremely experienced in software production, this is an important source of information. Because asking every employee at IBM what their opinion about a particular manner is, seems unrealistic. So, during questioning we must restrict ourselves to some resources:

- Handbook (IBM w3), representative for the company's standards and procedures. Much research has gone into construction and compilation of handbook, taking into account all facets of the company and their long history of software construction has led to the documentation of their experiences. The validity of the information in this document is unquestionable, however, the extent to which it is actually used may be.
- Warner Dijkhuizen, the contact person between us and IBM has strong interest in this project (and a successful product) and thus willing to help attain valid and precise information. He is very experienced with similar projects, and is representative of IBM due to his long work experience in the company (important aspects for reproducibility and integer). Furthermore, being a marketer, Warner is a people-person, understanding the needs of users and the issues of relevant people around them.
- Employees of similar projects, from within and outside of IBM, can be a valuable source of information to us. We have been introduced to people (at IBM or working alongside

IBM) of relatively high authority involved with similar and relevant projects (such as the 'safe internet' group), thereby increasing validity and objectivity. Open, semi-structured interviews can be used for distilling information where possible. Where physical distance is an issue, either structured interviews (for example by sending an e-mail) or short semi-structured interviews by phone resides to the possibilities.

Carers and Educators

Carers and educators, being the people who spend a lot of time with children, and most likely to know and understand children better than any other person, can be a valuable source of information. Being adults, and due to their experience with children, the information they yield is likely to adhere to most quality aspects. However, precaution must be taken on aspects of precision (they are not experts) and objectivity (too closely involved with children).

Preceding, a literary research has been conducted in search of some of the answers to the questions which carers and educators could give answers to. Because the resulting theoretical data is a generalization that was tested in other contexts, it may be incomplete and not adhere soundly in the environment in which our tool is set to work. So, open, semi-structured interviews are probably the most effective manner of distilling primary qualitative data from them, yet questionnaires would also work if a larger range is desired. But in both cases we must collect data from a range of both carers (parents, guardians, baby sitters) and educators (teachers, librarians, head of school).

Because our user group is international, it seems straightforward to select their carers, due to the international backgrounds, for interviewing. However, because Dutch children also belong to the user group, we must also interview Dutch carers and educators to make our sample of carers and educators representative for the whole group. Priority, however, goes to parents and carers who have or teach children in the ages of our user group. This need not be mandatory for the education. And upbringing of a child does not begin and end at the ages of our user group, but is rather an ongoing process and a child's experiences throughout his life will have an affect on him.

The data we receive from the interviews will be analyzed using inductively-based analytic strategies (see [22]), specifically, data reduction (includes summarizing and simplifying the data collected) and analytical induction (intensive examination of a strategically selected cases so as to empirically establish the causes of a specific phenomenon) will be methods used. See appendix A.3 for more information about the research to requirements of carers and educators.

Others

These are for example providers of internet locations and those socially involved such as governments, profit and non-profit organizations. They have published several documents and have posted many discussions on internet. Though most of these materials are not scientific literature, may be of lower quality and are most likely not very objective, their relevance and importance must not be overlooked. It can give us a wide range of interesting opinions and ideas. Each source must independently be checked for quality.

Secondary Scientific Literature

Secondary scientific literature generally adheres to all the quality criteria mentioned above. To assure that this is truly so, we apply a selection criteria of balance and agreement, trying to include most major ideas on important topics, conveying important facts and ideas on which researchers have agreed, and thereby concluding valid and generally accepted, and furthermore, precise information. The number of citations in other scientific documents to a particular author gives a rather simple yet good overview of their prominence and expertise in a particular field.

3.4 Results

This section describes the concluding results of the investigations completed in line of this project. We start by giving a short overview of the sources of data collected. Subsequently, in the data analysis, we summarize the results that answer our research questions, in the order they were posed. The answer to the first question includes a list of all requirements to which the product must adhere to, which can be used as a guideline for the design. It furthermore describes which requirements we plan not to implement.

3.4.1 Data Collection

The research strategy to gathering information has elaborately been described in Section 3.3.1. The following section describes the results and final outcomes of those investigations. The specific information gathered during the data collection phase is described in the appendixes. Table 3.1 gives an overview of that information. The following table gives an overview of that information:

Here follows a summary of the results of the data collection pertaining to each of the sub-questions.

Requirements

The precise method and results of the investigation into the requirements are described in appendix A.1. The stakeholder requirements (based on their sources of distress and required condition and capabilities) have been described separately per stakeholder in sections A.2, A.3, and A.4. The final result, the list of all stakeholder requirements can be found in section 3.4.2. To maintain overview, these requirements have been split up into three categories, functional, non-functional and business requirements.

Successful Design and Implementation

The investigation into successful design and implementations leads to a list of definitions, procedures and standards which, when followed, will lead to successful design and implementation of the product, which have been formulated as design guidelines (see section 3.5.2) for the following stage in the software engineering process. Details about the questions and aspects related to carrying out these interviews, along with their results can be found in appendix A.2.

Easy to Use and Support

The investigation into ease of use and support yields a list of characteristics to which the tool must adhere to in order to ensure adequate usability, and also comprise of a summary of the barriers that children experience when they are on the Internet (see section 3.4.2).

Search for Information

The investigation into search for information results in a list of characteristics to which the search results must adhere to (described in 3.4.2), and moreover a brainstorm of solutions to supporting children with search strategies. Which is to be chosen for implementation is determined in the design phase.

3.4.2 Data Analysis

In this section the outcomes of the investigations will be described for each question of section 3.2.1. Details about the different investigations can be found in appendices A.3 and A.4.6. As the research questions pertaining to 'successful design and implementation', 'ease of use and support', and 'search of information', are implicitly answered by the requirements, we will only discuss those here. The complete list of requirements distilled from all stakeholders are discussed below. Which of these requirements will actually be incorporated in the design and implemented will be discussed in section 3.5.1.

We furthermore define our tool as a whole to be successful if 80% of the defined requirements to be implemented have actually been implemented, and 70% of our interviewed stakeholders are satisfied with their experience with the tool. The measure of this success can be determined during the test phase.

The Requirements

The idea that children are masters of technology and can defeat any computer-related difficulty is a myth, children are incapable of overcoming many usability problems ([69]). Young people are not "just short adults" but an entirely different user population with their own culture, norms, complexities, and skills interacting with software in different ways than adults do, and having different motivations compelling them; a tool designed to aid children should adhere to and make use of their cognitive characteristics. With the emergence of children as important consumers of digital information, their role in the design of new technologies has become more prominent ([70],[78]). The degree and type of interactivity built into the portal should be based on the user characteristics. The content should be cool, but the design must offer high usability or kids will go elsewhere ([69]).

Each requirement described in this section describes motivation and applicability in the line of cognitive characteristics of the user group. It describes the requirements which are necessary to make an easy to use, and supporting tool for children in their search for information on the Internet. These requirements result from the investigations described in appendices A.3 and A.4.6. All of them are verified and complemented by literature research, which have been incorporated (and referenced to) in the justification of each requirement below. In general the final requirements list is merely the collection of all requirements found. Where contradictions were found, a compromise was made by means of discussion with the involved stakeholders. These compromises have been documented below.

Because the plethora of requirements that have come to light might burden the reader, we have distinguished among three levels of requirements, as is often done in practice [82]:

- **Functional requirements** describe the behaviors (functions or services) of the system that support user goals, tasks or activities.
- **Non-functional requirements** include constraints and qualities. Qualities (as defined from the point of view of the users perception, expectation and goals or need) are properties or characteristics of the system that its stakeholders care about and hence will affect their degree of satisfaction with the system. Constraints are not subject to negotiation and, unlike qualities, are off-limits during design trade-offs.
- **Business requirements** include high level objectives of the project, these have been described in the project plan. Only new requirements will be discussed in this document.

Each requirement is given an independent tag, to allow for the referral of a specific requirement in documentation composed throughout this project. The preliminary part of the tag refers to the type of requirement, with **F** indicating a functional requirement, **NF** a non-functional requirement, and **B** a business requirement. The second part identifies the (arbitrary) requirement number. For example, a requirement tagged F1 means that it is functional requirement number 1.

Functional Requirements
The goal of the user interface is to aid the user in understanding and expressing his information need and aid in the process of fulfilling this need. Studies have shown that the functionalities written in this section should be incorporated to aid the user with his task.

F1. Instructions / Help

In order to assure maximal usability, the program must have clear instructions, an on-line tutorial, and context-sensitive help ([59]). Speech output may be useful for help and extra assistance ([69]).

Contrary to adult users who avoid instructions and try to use web sites without having to read about what they are supposed to do, young users are often willing to read instructions, and research shows they even prefer to read a paragraph or so of instructions before starting a new game ([69]).

Children of ages 8 and up can follow fairly complex directions with little repetition ([101]). As a manner of instruction, use of rollover audio, animation and highlighting can be used to indicate functionality. Especially important for our user group is the use of short clear sentences for instructions; good examples of well written, non-condescending, instructions can be found in board games that are designed for kids in this age group, and may give a sense of good instructions that are age-appropriate.

As children primarily learn by imitation and practice [33], the help functions should show screen shots and example searches which the child can practice or imitate.

The interface should be intuitively easy to learn ([112]); children are likely to explore the software, clicking on things to see what they do, and often tend to repeat distracting activities just for amusement ([70])! Research shows that children were willing to ”mine-sweep”, traversing the screen with the mouse in search of clickable areas or simply to enjoy the sound effects that different screen elements played ([80]), and given their skill for picking up on navigation ([80]), explaining every detail of the interface in a separate help document is superfluous. Rather, given their willingness to explore, explanations of functionalities can be shown on-screen when the user hovers above a particular icon. In addition, by not making all elements obvious and explicit curiosity and thus interest can be maintained ([97]). Because of the nature of their exploration, the interface should provide clear visual messages, offer error prevention and simple error handling ([112]).

F2. Metaphor

Metaphors can intrigue the user and provide an attractive setting. A child who is emotionally engaged will experience a high level of attention, improved memory, and they may learn more about their emotional being or self. One of the best ways to increase emotional involvement is to link the content to a recognizable, constant character. Using a role-model metaphor in the design can reduce cognitive overload and help users utilize their own mental models ([109]). Expressive, colorful, and dramatic characters are often perceived to be more entertaining as well as helpful. It provides a unifying framework for design as well as facilitating the learning process by allowing users to draw on prior knowledge ([94]), and can present useful, but boring information in a fun easy to learn way ([75],[80]). Though a character should be very consistent in dialogue, their dialogue should not be limited too much. Occasional use of variations makes the character less static and more believable.

F3. Initiate Searching Process

A knowledge gap yields specific problems such as formulation of a query without knowing how to name what one is seeking. Two manners of helping our young user getting started could be:

- *Directory*: Possibility to choose from categories or defined subjects. With the Dewey Decimal Classification System, world's most widely used library classification system ([119]), well-defined and documented intuitive classification system part of many elementary school curricula around the world, using the same categories as those to categorize books in libraries seems obvious.

- *Example search results*: this will yield a short list of some previous search results of other users. Because users can acquire better knowledge and understanding of what they are seeking while glimpsing over relevant documents, which finally may make them capable of stating their need for information, we assume a certain correlation between different users and their search for information, and in such a manner, the user can profit from the queries formulated from other users. Searchers who have absolutely no clue what they want can view another users search example, and in such may help formulate their own information need ([59]).

F4. Formulation of Information Need

Describing their information need and determining which terms would best retrieve relevant information has shown to be problematic. In most cases the children's searches using queries are too general and result in the retrieval of too many irrelevant documents ([74]). Several navigation functionalities, whose use can be entwined and interleaved, could pose as a solution. Besides query input, category selection from a directory and relationship-map can be used to specify the information need. Furthermore, allowing natural-language as a query input and incorporating some form of visualization can help. Visualization can be incorporated by two main features (a directory listing and a relationship map), allowing both subject traversal and fine-tuning of the search description.

F5. Aid in Query Reformulation

Young children are concrete thinkers who seek information that can be found by using the exact terms set in the assignment ([35]), and encounter trouble with anything that is not an exact fit. Specific facts will be found easily by children only if their initial search terms happen to work, they will not find it straightforward to formulate alternative terms ([76]).

If too many or too few results are retrieved the system can help the user in query reformulation. When the results list is small, the initial query can be expanded to include spelling variations, term synonyms, or term hypernyms (is a kind of). In order to trim the results list, the user can narrow down the search by selecting a category or navigating through the relationship-map (including hyponyms and meronyms), in both cases the selections made will effect the input query and thereby also the resulting query.

F6. Following Progress of Search Process

The query created (due to directory or relationship-map selection) or modified (due to too few or too many results) in order to retrieve the result should be shown to the user. In such a manner the user can both better understand the retrieval result, and also learn, by example, how to create an appropriate Boolean query. Furthermore, though children have a better short term memory than adults ([69]) and are therefore more likely to recall which steps they have just made, for orientation purposes, an indication must be given as to the path that has been taken to get to the current location (i.e. the category selections that have been made).

F7. Giving Relevance Feedback and Understanding Retrieval Result

Searching for information is an iterative process, possibly even over several search sessions, the possibility to store previous choices and intermediate results must thus be offered ([104]), allowing the user to skip between different strategies and reducing working memory load ([112]). Informative feedback should be offered about the relationship between query specification and retrieval result, letting the user know why that particular result was retrieved. All accessible information about the current state in the search process should be shown including document title, a passage selection showing in which context the search terms (highlighted for emphasis) are used in the document, and the URL where it was found, to help users understand and predict the content associated with the link ([112]).

F8. Reversal of Actions

An ”undo” function should be included to allow for easy reversal of actions to return to a previous state and thereby reducing working memory load ([112]). The target age group is capable of understanding the notion of reversibility ([79]).

F9. Navigation and Search Strategies

Heterogenous social backgrounds, such as in our user group, show differences in seeking and understanding information ([94],[75]). Furthermore, levels of expertise in computer and information use ([94],[75]) and levels of domain knowledge ([115]) show differences in seeking and understanding information. The capability of formal operations, capable of thinking about ideas, considering various strategies and solutions involved in learning, memory and information processing without having to act them all out evolves around 11 years of age ([79]). Because this is the far end of our user group, we must aid the users in these aspects. The use of several different search strategies is mandatory for all the reasons mentioned above.

Information on the internet can be sought by a user using different strategies. An example is hypertext navigation, where the user jumps from one web site to the next through hyperlinks, is often used by beginner and novice users, however, it can prove to be disorienting and ineffective ([75]).

A portal can offer other navigation structures that have shown to be more effective:

- **Query input**:

  An information query, entered by the user, is matched against stored information. It has long been understood that adults have difficulty with the construction of logical Boolean-type search queries (conjunction, typically represented as AND, and disjunction, generally represented as OR), research has shown that children too, especially ages 5 to 12, have difficulty with Boolean concepts ([40]). However, under certain circumstances even children as young as three years have been shown to utilize disjunctive concepts to perform significantly better than chance, indicating potential ([40]). The user group generally finds it hard to generate search phrases, however more often uses multiple rather than single words in their queries ([74]). However, misinterpretation of disjunction and conjunction results in incorrect document retrieval. The interface must thus provide assistance with Boolean searches by consisting of several input fields:

  - results with the exact phrase (" ").
  - with all of the words (AND)
  - with at least one of the words (OR)
  - natural language input. Studies reveal that many children apply natural language in querying search engines ([62]). This requires an interface that translates the user-input into a more appropriate query that is to be used by the program internally ([63]).

  Our research has indicated that children have very much trouble with boolean searches, not understanding semantics, syntax or results. However, because of its power and use in adult search engines, we propose to embed a functionality which aids the user to input such queries in an 'Advanced Search', and for educational purposes, showing the generated query that is used to retrieve results.

  A help should be provided with tips on how to create an effective query:

  - explanation of boolean logic forms AND ('+'), OR and NOT ('-')
  - use of multiple words (or concepts) instead of one
  - use of quotes for phrases (words to appearing next to each other, in the same order, in a document)
  - use quotes to explicitly search for stopwords

  Furthermore, because children have a restricted verbal fluency and literacy level, the program should anticipate user spelling and typing errors. Some sources note the necessity for handling spelling errors ([59]) while others have shown that relatively few spelling mistakes were made when composing search statements ([74]). However, when no or few results are found, it must search for common spelling or typing variations (by means of edit distance) of the query typed in and add expand the original query with these terms. To aid users with their restricted vocabulary, in the case of few results being found, the query should be expanded with synonyms of terms used.

- **Directory**: A directory listing (or subject category index) makes use of a vertical (top-down) model or hierarchical model. The user starts at the broad end of a subject spectrum and step-by-step focuses in on more specific information by selecting categories. Option topics or directories are an especially good place to begin searching when the topic is broad, often leading the user to discover categories that help narrow the focus. There is some evidence that for a child using a Web portal in the exploratory phase of researching a school assignment, the most appropriate navigation structure is vertical ([75]). Related icons should be placed by the subjects to enhance comprehension ([80]).

- **Relationship-map**: A 3-dimensional bottom-up or relational map. Color similarity and empirical distance have a strong perceptual influence (Beck and Palmer, 2002) and can be used to represent the relationships and proximity of words or concepts which stand in relation to one another. The user can navigate through the map, automatically narrowing or expanding the input query at each step. Several word relationships can be used:

  - Hypernyms (given term is a kind of . . .): shows the relationship of the term to the hierarchical category that it is a part of.
  - Hyponyms (. . . is a kind of given term): shows the relationship of the term to other terms that belong to it as a hierarchical classification.
  - Meronyms (parts of given term): shows the relationship of the term to other terms of which it is built up of.

The structure used in the directory and relationship-map are identical. This type of browsing tends to be effective when users are pursuing less well-defined information objectives (Borgman et al., 1995). It places lower cognitive burdens on seekers, and offers possibilities of serendipitous discovery ([75]). Furthermore, studies of childrens searches provide evidence that their information seeking depends on the domain and topic knowledge ([70],[81]) and that children of this age group have a low information literacy ([75]) providing evidence that directory and relationship-map interfaces can be helpful.

Beyond 8 years of age, children are generally capable of handling opposite analogies easily ([101]), can relate rather involved accounts of events ([101]), are capable of performing multiple classification tasks and ordering objects in a logical sequence ([79]), categorical labels such as "number" or animal" become available at this age ([102]), and capable of concrete problem-solving ([79]); all of which poses evidence that they should be capable of using both directory listing and relationship-map interfaces. The task difficulty should match the user's current skill level, or rather be just a little bit demanding so the user is required to put extra effort into the task and thereby remain challenging and fun ([97]). Research indicates that geographic navigation metaphors works for kids; they like pictures of rooms, villages, 3D maps, or other simulated environments that serve as an overview and entry point to various site or sub-site features ([69]).

Research has shown that many users prefer a zooming functionality to navigate between different objects that are semantically related rather than jumping from one object to the next ([127]). The act of zooming from one object to the next, makes visually explicit where users are going and where they have been. Thus for navigational purposes, selections made from the directory listing or relationship-map must be shown as a path for orientation purposes and made clickable to allow for easy backtracking.

A portal for children should have the capabilities of providing both querying and category index tools. Experimentation has shown that children utilize both approaches ([77]), yet some studies claim children prefer browsing strategies ([74],[79]). This is most probably due to factors such as complexity of using multiple concepts in queries, where the search subject lies on the objective-subjective continuum, the relative prominence in the interface of the search box compared to the subject directory, and prior knowledge of the topic ([75],[76]). Practice (not training) has shown to yield little improvement in children's selection of search terms, choices between searching and browsing, and modification of strategies that are not working ([74]), hinting towards an improved interface to help overcome these difficulties. The three components (relationship map, directory listing, and input field) should interactively correspond to each other, giving the user an idea of where he is (map, links used from directory listing), how he got there (query input field) and allowing the user to switch between search strategies.

F10. Sitemap

Not pertaining to navigation over the web, but rather the navigation of our tool is the use of a sitemap for navigation within our site. Providing a visible sitemap facilitates site learning and encourages comprehensive exploration of a site. (Danielson, 2002)

Non-functional Requirements

NF1. Usability

What is important to the user is the time needed to learn how to use the system, time to achieve goal, chance of doing something wrong, and knowledge-halflife (how long before one forgets it). The tool must be easy to learn, mostly the goal must be achieved within acceptable time frame, and the probability of doing the wrong thing must be limited to a minimum.

Some of our research has indicated that kids often play on the computer in pairs. Where applicable and possible, this fact should be considered in design choices.

NF2. Presentation

On a high level, an effective and well-designed computer system generates positive feelings of success, competence, mastery and clarity in the user community ([112]). The interface should be transparent, not requiring or calling for explicit attention, and in this manner, leaving the user to concentrate solely on his or her work, exploration or pleasure. An attractive site will encourage children to stay with the site and return for subsequent visits. Such a site must make them feel comfortable, yet have the feeling they are in control. In essence, it should reflect a home. A place they feel comfortable and in which all necessary facilities are available, yet having a room they can design and use as they wish (obviously, to certain limits). Much research and experimentation has been done into usability aspects for building interfaces. Guidelines and best practices on how to design an effective interface follow:

1. *Font legibility*: Children who can read prefer larger font sizes, preferring 14-point Arial and 12-point Comic Sans MS font (Bernard, Mills, Frank and McKeown 2001). Highest legibility (for fast reading on the Web) is achieved with Arial. The most legible combination of colors are black on yellow, black on white, green on white, blue on white, white on blue, green on red, and red on green; other color combinations should be avoided.

2. *Icons*: Difficulties that children have with reading can be reduced by the use of 'visually meaningful icons', 'thoughtful cursor design', and the addition of features such as rollover, audio, animation, and highlighting (Hanna, Risden, Czerwinski and Alexander 1999). The users are capable of using symbols with ease ([79]), and at their stage of cognitive development, their understanding is tied to real, concrete, "touchable" objects ([34]).

   Younger children have difficulties with drag and drop activities (Joiner, Messer, Light and Littleton 1998) and [107], yet due to an improved eye coordination and motor control with age, their ability to point at targets, hold down mice buttons and integrate fine motor skills with perceptual skills ([69], [34]) should pose no special restriction to the icons for our user group. However, special attention should be paid to the clarity and mental models associated to icons ([111]). Children prefer animated icons, and such icons can offer more information than those with static representations [111] and (Baecker, Small and Mandler 1991).

   The best iconic buttons are easily recognizable, look clickable and are at least size of a Euro ([127]).

3. *Colors*: Children are attracted to bright colors ([76]), and can have a useful functionality for classification purposes. Gender differences "refer to social and psychological distinctions" ([31]) and must therefore be considered when making design decisions for an interface. In an attempt to make a web site that is not gender-specific, the design must find a balance between aspects often considered gender-related such light colors for girls and dark colors for boys ([113]).

   Concepts and links previously selected should be given a distinctive color in results list, directory listing, and the relationship-map to avoid iteration of the exact search.

4. *Language Level*: A child's reading and language understanding will have impact on interface design; words need to be chosen that are meaningful to the children. Young children prefer straightforward and simple language to allow them to understand the options available to them ([69]).

5. *Results lists*: Presenting 10 to 50 search returns per results page optimizes both performance and preference (Bernard, Baker and Fernandez, 2002) ([76]). However, cognitive research shows that lists, with or without scrolling, should comprise at most 7 items ([110]). Usually the user only wants a few documents anyway. Children rarely scroll pages and mainly interact with information visible above the fold ([69], and section A.4.4), thus the need for scrolling should be avoided, and at most 6 to 7 items should be listed. A selection from the document should be given with the query terms shown highlighted.

6. *Hyperlinks*: Because children often tend to re-activate hyperlinks previously visited ([63]), these should be given a distinctive color.

7. *Consistency*: Consistency among pages, language use, warnings, icons, and other representations should be maintained ([112]).

8. *Page layout*:

   - Users prematurely move their eyes to where they expect desired information on a next screen in anticipation, while the screen paints (Hornoff and Halverson, 2003). Moving animation should prematurely be placed at the location of most importance to attract attention while the user is waiting.
   - Users expect a back-to-home link in the upper left corner of the page (Bernard 2002).
   - Users expect Help icon in the upper right corner of a page (Bernard 2002).

9. *Home Page*: Children rarely search linearly, but rather repeatedly return to a ”landmark” page ([76]). Links chosen to other documents should be opened in a new page to ensure that Giggle be the landmark page and to maintain previously obtained results and the previous query (see section A.4.4).

10. *Fun and Interaction*: The primary goal of the portal is to help children in their search for information. Aspects related and influencing learning, such as motivation, attention, and concentration, are thus of great importance, allowing the users to not only attain information but learn new things. In order to learn anything, the subject must be paying attention, and incorporating features that attract attention rather than distract is mandatory to reach our goal. Richly mediated environments offer an appeal for children. Research shows that kids prefer and learn better with a rich combination of text, graphics, sound and animation ([94]). A fun interface can motivate users to explore the capabilities of the system, develop new skills, explore new domains, and learn new things ([97]). Children are very receptive to images, visual information, and sound effects; attractive screen designs based especially on effective use of color, graphics, and animation create a good first impression, make the site attractive to children and encouraged users to stay with a site ([69],[75],[76], see also section A.4.4).

    Kids like interactive content ([80]). Allowing kids to leave their mark on the web can increase interest and also promote subsequent visits to a site ([80]). Personalization such as alternative interfaces for novice and expert users, enabling frequent users to use shortcuts, change settings (such as background wallpaper, colors, layout or choose between metaphors), toggle features (such as an introduction message or the commentary by an animated character) ([112], [76]), or enable construction of a bookmarks list with favorite links, will allow the user to adapt the site to his or her liking, allowing the user to control aspects that are meaningful and important to the user for optimality ([97]), and thereby making it more likely that the kid will return to the site again and again.

    Besides wanting to provide users an enjoyable experience, by implementing game-like features one can increase motivation, stimulating and encouraging users to continue using the product. Sometimes kids just want to have fun and play. Playing with a computer can enhance comfort with the technology and encourage kids to branch out and experiment with other applications. Play is education too, and this calls for an incorporation of some interactive games, collaborative storytelling, polls, and trivia tests in the design. Because children of this age group typically have a short attention span, activities lasting longer than 15 minutes should be divided into levels or sections so that a child can complete one and be able to stop with some sense of completion ([80]), or restrict content to short trivia and games.

11. *Time and Date*: Our investigation has shown that some children would like to see the date (and possibly the time) on the screen. Obviously, a clock fun and interaction would be the best.

12. *Disturbing features*: Dialog-windows, popups and banners provide information that is unasked for by the user. Most users find them annoying, their appearance should be obstructed. Furthermore, their content is often difficult or impossible to determine due to incorporated graphics which can't be interpreted automatically.

NF3. Relevance of Retrieval Results and Filtering

Requirements analysis has indicated that the primary aim is to increase precision with respect to retrieval results. That is to say, in the trade off between possibly retrieving inappropriate document or retrieving a smaller set of documents, the general preference is for the latter; the child's safety has a higher priority than a wide range of resulting documents. We must thus determine which documents are favorable as apposed to inappropriate or unfavorable, and even more so, how to determine this.

A question we pose ourselves is whether or not to filter content based on appropriateness for the age group. On one hand some stakeholders may favor the idea, yet it is also brings up much criticism. Some argue that filtering services are really a high-tech form of censorship ([34]). Libraries which stand for the preservation democratic society by making available the widest possible range of viewpoints, opinions, and ideas are restricted in their mission by filtering services ([34]). Filters may also provide a false sense of security, compelling users to rely on software instead of personal judgment or as a replacement for parental supervision. Such technology is merely not foolproof, failing to capture all the arguably bad sites while sometimes blocking those that are arguably good. Because of the Internet's unstructured, unregulated and international nature, there is no assurance that any software will block access to all materials that fall within selected blocked categories. It is also possible that the filter will block information that it should not. Research shows that filters fail to block objectionable content 25% of the time, while on the other hand, they improperly block 21% of benign content ([121]). Over-inclusive filters cause particular damage to any content dealing with gays, safe sex, and left leaning political groups. We pose ourselves the question whether we wish to meddle ourselves in the discussion about what is appropriate and what is not, which action enforce people's rights, and which evade them, and how to ensure and guarantee accuracy and precision in retrieved results.

However, our research has indicated that many stakeholders find this an important aspect and a characteristic that must be implemented in the resulting tool. Furthermore, we can justify that unsafe and inappropriate documents do not belong to the set of relevant documents the user wishes to retrieve. Some form of filtering will be incorporated, however, full protection will not be guaranteed.

We distinguish between different sets of documents being benign (as opposed to harmful as described in the Requirements Analysis) and being relevant to the user according to the information need. Here follows a description of the different sets of documents and how they can be detected (See also Figure 3.2):

- **White-listing**: Documents with benign and safe content with respect to our user group.
- **Black-listing**: Documents with inappropriate (unsafe, unfit, obscene or harmful to minors [71]) content. Our research has shown stakeholder demands to filter on the following aspects:
  - adult content: sexually explicit graphic descriptions or images;
  - child pornography;
  - hate sites: advocating bigotry, hatred, or discrimination;
  - graphic violence: violent images, language, bomb-building, etc.;
  - criminal activity: promoting illegal activity;
  - other offensive sites: alcohol or drug abuse, or promoting suicide and violence;
  - incomplete or poorly maintained sites;
  - pop-ups, advertisements, and banners;
  - Unprocessable sites: Prevent any information to be shown that does not allow processing or content determination (such as encrypted information, or websites not allowing automatic access);
  - Prevention of exchange of personal information;
  - Prevention of online shopping and banking;
  - Prevent harmful doing to software and hardware: viruses, automatic installations (such as toolbars and programs).
- **Gray-listing**: Documents of which the listing category has not yet been determined, but their content could potentially be inappropriate.
- **Favorable document**: A document that adheres to the specifications described in the requirements document, such as reading levels and content topics. Favorable documents must adhere to the following properties:
  - **Adequate Literacy Level**: Children require comprehensive language that is accessible (short sentences containing few difficult words), long texts written above the reading level of the reader are problematic for young children. Children of our target age group should be reading with considerable ease and they are capable of understanding more vocabulary and concepts than they may themselves express ([101]). Retrieved documents may contain complex and compound sentences ([101]), however, should be targeted under or slightly above their reading age ([103]). Several reading formulas exist that are capable of giving an indication of reading difficulty of a text. Students reading materials tuned to their reading level report competence, confidence and control over the text, whereas when a text measure is greater than a reader's measure, comprehension drops dramatically, and the subjective experience is one of frustration, inadequacy and lack of control ([103]). See appendix A.4.6 for more information.

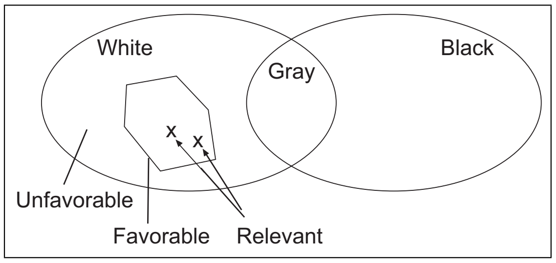


Figure 3.2: Definitions of document types.

  - **Listings**: It is a subset of white listing documents, and absolutely not an element of either black or gray listing.
  - **Content and presentation**: Statistical analysis on the corpus of a children's search engine can give rise to important information on what children find relevant, such as which concepts are common or dominant, (mean, mode and median) document, sentence and word lengths, or number of pictures, could give a rather good indication of what types of document children find relevant.

- **Non-favorable document**: Any white-listing document that is not a favorable document according to one or more of the specifications in the requirements document is non-favorable. All black and gray-listings are also non-favorable documents.
- **Relevant document**: A document capable of answering the information need posed by the user. This may be an element of a favorable document set or a gray-listing.
- **Irrelevant document**: Any document incapable of answering the information need posed by the user. We propose that all documents belonging to the set of black-listing also belong to this set.

For retrieval, the aim is to only retrieve favorable documents. However, because gray concepts are unknown concepts, they may occasionally also contain a relevant document and can potentially be retrieved. In the document ranking, favorable documents should attain a higher ranking than gray documents.

Furthermore, statistical analysis on the corpus of a children's search engine can give rise to important information on what children find relevant. Several aspects such as which concepts are common or dominant, (mean, mode and median) document, sentence and word lengths, or number of pictures, could give a rather good indication of what types of document children find relevant.

After launching the system, a lot of information can be derived from the children using it to improve the retrieved results. By paying attention to the way in which the system is used, valuable information can be distilled out of the documents retrieved and those the users indicate to be relevant, such as dominant concepts, document types and document formats. This information can be used as an indication to what users find favorable documents and can be incorporated to fine tune the retrieval results. Also, by observation of user query formulations and preferred search strategies can also be distilled.

NF4. Safety Warning

Several sources have indicated that the user is to be explicitly warned of the dangers posed when leaving the Giggle site ([49],[58]), in order to avoid providing a false sense of security. Thus, when the user chooses a resource outside of the site, a warning must be posted explaining the possible dangers: "You have chosen a resource outside the Giggle site. We have provided links to resources that we hope will be helpful to you, but remember, Giggle cannot vouch for their content. Thanks for visiting!". To protect personal information, when collecting a user name and password, the user must be alerted to use a fictitious name and a new password that is not otherwise used for any other purpose: "Don't risk your safety! Please never give out personal information to strangers or companies. Be aware that almost everyone you meet on internet is a stranger! Never give out your own name, use a nick name. Never give out a password you use for anything else, make up a new one!". Furthermore, all user information must be encrypted during transmission. In order to recover a lost password, at the time of registration the user is to give the answer to a question only they know an answer to. This can be used to authenticate the user for recovery.

NF5. Hardware, Software, and Formats

The tool must be compatible with existing hard and software that the children use. According to our research, this means it must be taken into account that the tool must run on a Microsoft Windows 2000 operating system (or higher). All features should be functional using Microsoft Internet Explorer and Mozilla browsers. Furthermore, functionality should not be dependent on the capability of the computer to save cookies, install programs, or use of pop-ups, as this is often disallowed. The tool must be capable of running alongside an existing family filter.

Furthermore, several different format types are consulted in search of information. HTML is most commonly used and read by our users and legible on all machines, followed by Microsoft Word which is also commonly used and installed on almost all user-machines, and PDF which less commonly used, but often retrieved with general search engines. Other format types such as Microsoft Excel, PostScript, and Microsoft Powerpoint are less commonly used, seldomly retrieved and generally cannot be read without prior installation of other programs. The tool should be capable of retrieving documents most frequently consulted and able to be viewed on general user machines.

NF6. Reaction Time

Like the busy corporate executive, children with homework assignments have little time to spare for leisurely interactions, and are just as intolerant of delay irresponsiveness as adults ([74]). In contrast, when the portal is used for entertainment the children would welcome high levels of interactivity and the consequent control as well as guidance that they believe this gives them ([75]).

Kids have a shorter (and more selective) attention span than adults, lack patience and capacity to concentrate ([34]), and want an immediate reaction when they click on something ([69]). Animation can be used to distract children while they have to wait for the system to perform while giving feedback by showing that the input has been recognized and that the content is coming, and as a good tool for showing that the state of an object has changed.

Business Requirements

The business requirements have elaborately been discussed in the Project Plan document. This section only includes new requirements that have come to light during the requirements analysis phase. We emphasize that the complete list of business requirements is the collection of all business requirements posed in the Project Plan and those that follow here.

B1. Awareness

With IBM's focus on involvement in the community, their investment in this project is made worthwhile by having people aware of their investment, and thus they stress the necessity to increase awareness about this project.

Action has been taken to increase awareness and acceptance of our product within IBM, which we hope in turn will increase awareness outside of IBM. Visible connections have been made with current initiatives as well as contact with their project managers (such as the EMEA Corporate Community Relations Programme Manager in charge of 'Safe Internet' group, and Programme Manager for Education Outreach) reaching as far as the Great Britain and the U.S.A., and information about the process and progress of the project has been posted on IBM's intranet, the On Demand Community site.

Furthermore, awareness within the community must be raised. As kids often tend to visit the same sites and in order to discover new sites they need help from parents, this may be a potential gateway. IBM's safe internet group focusses on raising awareness about internet safety by educating parents and teachers. We have made contact with the manager of this group, raising awareness and having found much common ground between projects, a good initiative to use IBM's current projects to let the world know what we are working on. Furthermore, advertising surprisingly ranks high as a tool to discover new web sites ([57]). We have also promoted the tool at an international school during a course taught by us about safe internet, and written an article explaining the need and focus of the project in the school newspaper.

B2. Multi-linguistic

In line with IBM's global business, a product with a global span is desired. As such, the capability to include user groups as those who are native in English, Spanish, and Dutch is desired. Because children are not capable of adequately understanding another language than their native language (even though they may understand a few words), this means the interface, query input and retrieved documents must be in their native language. Obviously, there must be a functionality to set the language of the user's choice.

3.5 Conclusion

The goal of the requirements analysis phase was to determine the relevant requirements posed in order to successfully design and implement an easy to use tool that supports children in their search for information on the Internet.

In short, our investigation has lead us to the following conclusion. In order to support a child in their search on internet, they not only seek a search engine that will find information appropriate to their needs and interests, but they need a tool they feel comfortable with and attracted to, a tool that has been specifically built for children. As the internet brings to the world a new dimension, children seek a place to call home in this new world. A place they feel comfortable, safe and at ease, a place they start the day off and a place to come back home to, a place where they have their own room that they can design organize as they wish in a house having all the facilities they need. They need a website like Giggle that they can mark and experience as their home in the cyberworld.

The complete conclusion of this chapter comprises of the answers to the specific research questions posed in section 3.2.1, out of which we have distilled the conclusion in the previous paragraph. However, the answer to the first question is merely the complete list of requirements, which implicitly includes the answer to the last three questions. For the complete list of requirements we refer to section 3.4.2 in which a detailed description of each has been given.

3.5.1 Prioritization

The list of stakeholder requirements included in section 3.4.2 is merely the collection of all requirements found. Where contradictions were found in requirements from different stakeholders, a compromise was made by means of discussion with our client and involved stakeholders in order to solve these.

As the client, IBM is the most important stakeholder in this project. It is IBM's wish, in case time constrains the project, to have some prominent features completely and correctly implemented, rather than many (interesting) features that cannot be implemented. Certain key features will be implemented one by one, ensuring that the most important features to be completed, and thereby leaving out less important features if there is no time. This also ensures that the product does not show features that do not work, such as an icon which has no functionality. Obviously, dropping requirements poses an extreme restriction on our product. The product can simply not be complete if not all requirements are adhered to. However, with instruction from our client, we have determined to drop some requirements, implying restrictions on the completeness of our product, to ensure that at least most of the important requirements are taken into consideration. Where time allows, or the project is continued in the near future, the remaining requirements can be implemented.

We briefly verify to ensure that the developmental model we have chosen is adequate for this approach, the incremental implementation of prioritized features. As explained in the Project Plan, we have chosen for an incremental implementation model (the waterfall model) for the project as a whole, which does indeed allow for such an approach.

This section describes which product aspects have priority over which other features determined by their relative importance. In case we have not enough time to realize all the requirements, those of the secondary list will be dropped first. The choices made have been listed in tables 3.2, 3.3, and 3.4 and where necessary, are justified in the sections that follow.

Primary Requirements

The requirements described in this document has lead to the following list of basic functionalities (with their corresponding characteristics) that must, at minimum, be implemented:

- **Instructions/Help (F1, F3, F9, NF1)**
  On-line tutorial explaining all functionalities. Includes a guided tour of all options and a complete search example from initialization to finalization to show all intermediate steps of the search process. Furthermore, upon hovering the cursor above an option, context-sensitive help will be given explaining in short the purpose and use of that functionality. This function will also give some tips about how to effectively formulate a query.

Nr.	Description	Feature	P/S/D
F1	Instructions/Help	On-line Tutorial	P
		Context Sensitive Help	P
F2	Metaphor	Character	P
F3	Initiate Searching Process	Directory	P
		Example Search	P
F4	Formulation Information Need	Keyword query input	P
		Directory	P
		Relationship-map	P
		Natural-language query input	D
F5	Aid Query Reformulation	Query expansion (synonyms/spelling)	P
		Specification category/rel-map	P
F6	Following Progress Search Process	Modified query	P
		Path chosen (clickable)	P
F7	Relevance Feedback	Store intermediate results	P
		Relationship query+result	P
F8	Reversal of Actions	Undo/Step back	P
		Redo/Step forward	P
F9	Navigation and Search Strategies	Keyword query input	P
		Directory	P
		Relationship map	P
		Corresponding components	P
		Advanced search	S
		Query input tips	P
F10	Sitemap	Map	D

Table 3.2: Summary of **P**(rimary) and **S**(econdary) and **D**(iscarded) functional requirements.

Nr.	**Description**	**Feature**	**P/S/D**
NF1	Usability	Easy to learn/use	P
		Multi-user	D
NF2	Presentation	Interface guidelines	P
		Make Giggle start page	P
		Login and personalization	P
		Block popups, banners, ads	P
NF3	Relevance and Filtering	Filter inappropriate content	P
		Filter on reading level	P
		Filtering on other aspects	S
		Promote favorable concepts	P
		System learning from user input	S
NF4	Safety Warning	Educate user about safety	P
		Encryption user information	P
NF5	Hard and Software	Windows 2000 and up OS	P
		IE and Mozilla browsers	P
		Existing filters	P
NF6	Reaction Time	Adequate performance	P
		High performance	D

Table 3.3: Summary of **P**(rimary) and **S**(econdary) and **D**(iscarded) non-functional requirements.

Nr.	**Description**	**Feature**	**P/S/D**
B1	Awareness	IBM	P
		School	P
		Community	S
B2	Multi-linguistic	English	P
		Dutch	S
		Spanish	S

Table 3.4: Summary of **P**(rimary) and **S**(econdary) and **D**(iscarded) business requirements.

- **Metaphor (F2, NF1, NF2)**
  An animated character who gives advice (tip-of-the-day) and whose entertaining livelihood makes the page interesting and dynamic.

- **Query input (F4, F5, F6, F9, NF1)**
  Standard method of query input field with accompanying 'Find' button to initiate the search process. The input can constitute of one or more keywords, including the possibility to use quotes and boolean syntax (AND, OR, NOT). Assumed is that a sequence of words is a reference to resulting documents containing all the specified words. Includes a stemming, spell check, and synonym function which gives suggestions for alternative spellings and adds variations of keywords to the input query (query expansion). It is furthermore not case sensitive. After query modification, the field is to be updated to present the modified query (used to retrieve the current results) to the user.

- **Directory (F3, F4, F5, F9, NF1)**
  High-level intuitive directory (accompanied with icons for clarity) allowing for selection by mouse to (further) specify information need.

- **Relationship-map (F4, F5, F9)**
  3-D map indicating relationships (synonyms, hypernyms, etc.) between the input keyword and other words in documents predicted to be relevant. Words in the map can be selected by the mouse to (further) specify the user's information need. Interacts with corresponding directory and query input field components.

- **Query modification (F5, NF3)**
  Query modification or expansion can be done by the system to include stemming and spelling variations and synonyms. Selections made from directory, relationship-map, advanced search or query input field specify and thus modify the query that is to be used. Any additional information or modifications by the system, or during user reformulation, which alter the input query will be shown in the query input display which always indicates the query used to retrieve the current results. Furthermore, user relevance feedback can be used to reweigh, and thus modify, the query to promote documents containing relevant and favorable concepts.

- **Results list (F7)**
  Shows a list of approximately 6 retrieved docs, including a short summary with keywords (or synonyms thereof) highlighted and a link to view the document. Relevant documents are temporarily stored in the list as intermediate result, documents marked as irrelevant are discarded immediately.

- **Result document (F7)**
  A document chosen from the results list is opened in a separate page. The keywords relevant to the query (including stemming/spelling variations and synonyms) are highlighted. The user indicates interest in the document (option to temporarily store in list or save to file) indicating relevance, or option to discard the document (remove from results list) indicating irrelevance. In both cases information is extracted from the document and the query is reweighed resulting in a new results list.

- **Reversal of Actions (F8, NF1)**
  Navigation buttons undo/step back and redo/step forward to allow simple reversal of

actions. It must be able to undo all actions to avoid disastrous effects as a result of learning by trial and error.

- **User safety (NF2, NF3, NF4)**
  A filtering mechanism is to block popups, banners, advertisements, inappropriate content and reading level. Where appropriate (when leaving our site or inputting user information) the system warns the user of the dangers of the internet and precautions to take. All user information is encrypted during transmission.

- **Personalization (NF2, NF4)**
  Capability to make the Giggle website the starting page of the browser. In order to make use of more functionalities the user must authenticate himself. (Encrypted) Login for existing users and registration for new users, allowing personalization functionalities such as editing constituents of a favorite links list, changing the appearance of the screen (background color), toggle features (tip-of-the-day or metaphor), changing password, making our site the browser's starting page, or indicating the interface and search language (English/Dutch/Spanish). Also allows the user to save and load a search state (results and query input).

- **Links list (NF2)**
  List of links to informative sites (dictionary, encyclopedia (encarta.msn.com), atlas, math help (www.purplemath.com), translator (www.worldlingo.com)). Also includes a list of links to entertaining sites (games, jokes).

- **Time and date (NF2)**
  Realtime clock showing time and date in an entertaining fashion.

- **Hardware and Software (NF5)**
  Product must be capable of running on Microsoft Windows OS versions 2000 and higher. All options must be functional using Internet Explorer or Mozilla browsers, and capable of working alongside existing filters.

- **Performance (NF6)**
  The system must have an adequate performance. Though not the focus of this project, reasonable precautions will be taken to keep reaction time to a minimum.

- **Awareness (B1)**
  We will make effort to increase awareness of the product within IBM and in schools.

Secondary Requirements

The requirements stated in this section are nice to have, will increase the popularity and improve the tool, but are not necessary to have a minimal functional product. The requirements below are ordered on importance, most important first.

- **Advanced search (F9)**
  In addition to the traditional query input field, the user can choose to attain help to input a more advanced query search. To aid with the creation of Boolean query input, a selection of text fields are given in which important terms can be filled in (representing the boolean functions AND, OR, NOT, complete phrase input). It also

includes instruction on the semantics and syntax of boolean inputs and tips on how to effectively formulate a query. The software will translate these fields into an adequate query. The generated query will be shown to the user in order to educate the user of the capabilities. Though many search engines allow the 'NOT' keyword to be used, it poses several problems. By solely indicating that a term may not appear in a document would result in retrieving almost every document in hyperspace (i.e. the query "NOT bike" will result in all every single document as long as "bike" does not appear in it), and is probably not what the user wants [106]. Furthermore, research has shown that users often misunderstand the semantics of the keyword "NOT", resulting in misuse. However, some users have indicated that this functionality is useful and wish to have it appear in the final product. We have decided that if he cost of implementing this feature does not weigh against its benefits it will not be incorporated in the implementation of the product. However, because this decision will be made at a latter point in time, it will be incorporated in the design.

- **Multi-linguistic (B2)**
  Due to linguistic differences, difficulties that must be overcome with respect to document retrieval, and the amount of existing research available, we have chosen to prioritize linguistic capabilities as follows:

  1. English
  2. Dutch
  3. Spanish

  The web site will first be made to deal with English documents, queries and users. Upon completion of a functional prototype in English, and if time allows, we will continue with Dutch and thereafter incorporate Spanish facilities.

- **Document Format Types (NF5)**
  Not all format types allow for simple manipulation should as text extraction and the highlighting of queried terms. Because this requirement has shown to be a must according to the requirements, it is our goal to ensure that this capability is to be implemented. To overcome technical barriers that cost a lot of time, we have decided to primarily focus on document formats which are easily manipulated and furthermore most often used (according to our research), and after completion of the requirements, continue to other format types. This results in the following prioritization:

  1. HTML: most commonly used and read by our users, furthermore rather simple to extract and manipulate text.
  2. Microsoft Word: also commonly used and installed on almost all user-machines. Text extraction and manipulation rather simple.
  3. PDF: less commonly used, but often retrieved. Text manipulation is a barrier that must be overcome.
  4. Microsoft Excel: less commonly used, seldomly retrieved, program installed on almost all user-machines.
  5. PostScript: Sometimes retrieved, however cannot be read on merely every computer without prior installation of a (free) viewing program. Keep in mind that not

all users have the permission to install a program on the computer they use. Thus, not all users are capable of reading this format. Text extraction and manipulation is a large barrier that must be overcome.

6. Microsoft Powerpoint: less commonly used, seldomly retrieved, program only sometimes is installed on user-machines, is furthermore very expensive and thus not all users are capable of reading this format.

- **Relevance and Filtering (NF3)**
  Other filtering requirements such as recent modification date, quality of source, and interpretation of pictures, are nice to have but are not a primary requirement. Furthermore, a system that is capable of monitoring user input and using that and statistical analysis as a basis for learning is a preference to yield results of higher precision, yet not mandatory for the system to work.

- **Awareness (B1)**
  The world is a big place. The two of us have put much effort into increasing awareness, yet have not found the time or possibility to raise awareness in the community. We hope our efforts of raising awareness at IBM and in schools is adequate in this preliminary stage and that we have focussed on people who have a communicative role, thereby reaching more people than those we have merely spoken to.

Dropped Requirements

Not all requirements defined as useful by one or more stakeholders can or will be implemented. This section describes the features and the reason for not implementing them. Some of the requirements that will not be implemented can serve as a good starting point in contemplating the expansion of tool at a later stage, in this case it has been indicated.

- **Multi-user (NF1)**
  Children often work or play on the computer with a friend [70], and a proper design must consider this. However, our tool is made to stimulate and encourage children to find information on their own, and at this particular time, we have not paid attention to the possibility incorporating collaborative activities.

- **Text Interpretation (NF2)**
  To increase legibility and interface consistency, and assure accessibility in spite of document type (not every child has a Postscript viewer installed on the computer, and Microsoft Word documents cannot be opened under a Unix or Macintosh environment), the documents retrieved should be interpreted (by text-extraction or Optical Character Reading) and should be presented on screen in the same font type, size and style as the rest of the interface. Words involved in the query, directory or relationship-map should be highlighted in the document to show why that particular document was retrieved and which section of the document is most relevant to the query. This requirement will only be implemented for html and flat text documents, for all other formats this requirement has been dropped due to a lack of time. If time lends it, this extra feature would be interesting to implement.

- **Safety Precautions (NF3, NF4)**
  Though it is of interest to protect the children from particular dangers, it is impossible

to protect them from all hazards and dangers. Because they are unrealizable within the scope and time constraints of the project, some specific aspects with respect to protection have been omitted:

– *Exchange of personal information*: Many stakeholders have shown an interest in the protection of personal information such as address, telephone number, parents work address/telephone number, or the name and location of the child's school, pictures, passwords, or credit card information ([44],[51],[53],[54]). Probably the simplest way of achieving this protection is the omission of web sites that allow or require the input of this type of information; however, this approach also has the effect of dramatically restricting the capabilities of the user. Though the amount of attention paid to the topic does make it a good feature when contemplating the extension and expansion of the web site, this type of protection does not contribute to the primary goal of the search engine (which is to support the search of information) and unrealizable within the scope and time constraints of the project.

– *Chatrooms*: Many sources have shown that children spend a lot of time online chatting, they enjoy interacting with other people and they can feel socially connected and foster friendships in this manner ([80]). However, as mentioned in the previous paragraph, the exchange of personal information to strangers and the possibility of receiving inappropriate messages, pose a danger to children. One out of every ten children is confronted with sexual allusions during chat sessions, and two are teased or scolded at [50]. 65% of all children on internet have exchanged their e-mail address with another chatter, of which 10% later experienced that this had very negative results (based on research by Kidsweek and the Nationale Kinderkrant)[50]. A possible approach to protection would be to filter the information exchanged. However, this approach can be debated to be unconstitutional and an invasion of privacy. Chatting also has positive aspects, it is an additional communication medium and can be used for educational purposes, such as contact with foreign schools or children. Furthermore, some relatively safe chat programs exist, like MSN, because it is a 'closed' chatbox with, for children, clear capabilities to block out and become invisible from unwanted chatters [14]. Due to the controversy that it calls and the problems associated with it (i.e. Children's Online Privacy Protection Act (COPPA) which is related specifically to the safety of our user group has been regarded by many as unconstitutional([34])), we have decided that ensuring the child's safety while chatting is essential but unattainable in the short time span that we have and thus falls beyond the scope of this project.

– *Graphics*: Though several stakeholders have shown interest in the filtering of graphics and images, this is a very complicated task for which we do not have enough time to thoroughly investigate. We can merely filter graphics by their content on their file name and not explicitly the contents of the picture.

– *Soft/Hardware Protection*: The prevention of intrusions of viruses, worms, trojan horses, automatic installation of programs and toolbars, and other harmful (or less harmful) software, is a feature several stakeholder groups wish for. As the owner of the computer is responsible for adequate protection, it falls beyond the scope of our responsibilities with respect to information retrieval and thus this project to implement these aspects. Minimal protection could be verifying that downloaded programs have a valid certificate, that the identity of the software publisher matches the certificate, and that the certificate is still valid. Furthermore, blocking files that contains an ActiveX control or Web browser add-on that doe not have a valid digital signature preventive measure to block toolbars, stock tickers, video, animated content, which are sometimes capable of collecting information from a computer in ways the user might not approve of, possibly damaging data on your computer, installing software without your consent, or allowing someone else to control your computer remotely. A digital signature is a way to electronically mark a file with information indicating the publisher name and that the file has not been changed after it was signed, tampering makes the signature invalid.

- **Natural Language Query Input (F4)**
  Though it sounds attractive and useful for children, query input in the form of complete natural language sentences, followed by interpretation and translation into an adequate query to yield results, is both impractical due to word-ambiguities and is not used by the focus group. Our investigation shows that children merely do not want to laboriously type in entire sentences, risking spelling and typing mistakes. Furthermore, they are all acquainted with using short query formulations. Our efforts in this field will most probably be superfluous.

- **High Performance (NF6)**
  Dropped reaction times issues with respect to high performance. Concentrating on hardware systems and database issues to fully maximize and optimize performance falls beyond the scope of making a pilot prototype. Furthermore, these issues rely heavily on hardware, a component likely to change when the pilot is implemented for real use.

- **Site Map (F10)**
  Though research indicates that each site must have a site map, we have explicitly chosen to leave this option out. The reason being is that the website has minimal number of pages, the user can merely not get lost. In our case, the site map poses no advantages.

- **Results Translation (B2)**
  As the World Wide Web is an international medium with documents posted in all sorts of language, breaking the language barrier with a translation feature would be a very desirable function. Because automatic translation facilities have shown to perform inadequate and inaccurate translations leading children to confusion, furthermore text interpretation falls beyond the scope of this project, we have chosen to not make use of this facility to translate results. However, this is a wonderful feature when contemplating the expansion of Giggle, momentarily the focus is only on the retrieval of documents in the language input (English, Dutch or Spanish).

3.5.2 Design Guidelines

As IBM had a long history and much experience with developing software we have learned about and adopted their finest best practice methods and used them to plan the remainder of the software engineering project. Section A.2.4 describes which best practice methods have been advised to us for our project. As these methods do not pertain specifically to the requirements analysis phase they are discussed in the documents of the phases to which they adhere to. For example, specific aspects pertaining methods chosen for the design of the tool and justification hereof can be found in the design document, and those pertaining to testing can be found in the test document. However, because the requirements analysis phase and the design phase are adjacent, a short introduction to the design phase is wise at this point in time and follows below.

The resulting list of prioritized requirements hints towards a specific design approach. As explained in the project plan, we have chosen for an incremental approach to implementation and testing. This allows us to create a partially functional prototype based on the most important aspects, without having to wait for full functionality of some details. Coinciding with IBM's best practice methods for software design, this implies a modular design. That is to say, during the design phase we must indicate which functionalities belong together in a module. Each module will then be implemented and tested one by one, and be made in such a manner that they can be integrating coherently resulting in a functional and consistent product. Modules should be constructed to be rather independent, alteration of one module should not impact another. Communication with the database must be encapsulated by one module only to restrict impact upon adaption of the database.

To ensure a correct and complete translation of requirements into design and continuing on to implementation, best practices such as the use of UML (including detailed class diagrams, collaboration diagrams, sequence diagrams, state diagrams, and activity diagram) to ensure correct functioning, and ORM to ensure a correct and complete description of the database.

We must keep in mind that all primary and secondary requirements may be implemented at one time or another, and to ensure that we don't make design choices that might undermine some requirements, all requirements must be incorporated in the design. Thus, the approach to design is not incremental. However, also here we make a prioritization. The primary requirements will be incorporated in full detail into the design, the secondary requirements will be taken into account on a higher scale, but not worked out on a detailed level.

3.5.3 Evaluation

Though we were obviously unable to research and investigate every relevant person on the globe, we had to make decisions as to whom we would investigate. In determining this we considered aspects pertaining to the quality of information, such as validity, reproducibility and precision. In our opinion we were able to gather a rather precise and complete view of aspects pertaining to the problems in question. At least, for the amount of time and manpower that is available.

Furthermore, in order to verify the conclusions that we have made, we have let several people read and review preliminary versions of this chapter and comment on it. As such, in this final version, the comments of our client contact at IBM, an IBM software programmer, two parents and one middle school teacher on preliminary versions have been incorporated. Conclusively, these people have agreed, as far as they are capable of determining such, that the aspects described in this document are valid and complete.

Chapter 4

Design

4.1 Executive Summary

This chapter describes how the implementation of Giggle is to take place. The specifications of the Requirements Analysis (see section 3.5.1) have been translated into formal models which form the basis for the implementation phase. These formal models express the functionality of the system to be made, in a manner that can not only be understood by both the programmers that are to code and implement the system, but also the managers and supervisors of the project. The models furthermore describe a breakdown of the system into multiple components which can each be coded and tested separately (see sections 4.4 and 4.5). This chapter also includes the first version of the user interface (see section 4.6), and the results of preliminary usability testing on users (see section 4.7).

4.2 Introduction

The goal of this chapter is to describe the design of Giggle. The design phase is just one of the many software engineering phases in bringing the tool to its existence. The Requirements Analysis, which was completed prior to initiation of this phase, is the input of the design. The output of the design is a complete specification and plan indicating how the implementation phase is to proceed. Creating a correct design is imperative to designing a successful product. We must thus ensure a correct translation from input to output.

During the design phase, all aspects and (primary and secondary) requirements mentioned in the Requirements Analysis document are to be translated into an implementation plan. To ensure that this happens correctly, the requirements and use cases will be modelled using proven well-tested methods, a standard used in all IBM software development processes. These formal techniques ensure correctness of the description, reduction of ambiguities and inconsistencies, validation of the architecture the architecture and an increased readability of the description. The result is a specification which can easily be read and translated for implementation purposes, which is to occur in the next developmental phase.

As the client, IBM is the most important stakeholder in this project. It is IBM's wish, in case time constrains the project, to have some prominent features completely and correctly implemented, rather than many (interesting) features that cannot be implemented. To ensure this is the case we have chosen for an incremental development model. Certain key features will be implemented one by one, ensuring that the most important features to be completed, and thereby leaving out less important features if there is no time. This also ensures that the product does not show features that do not work, such as an icon which has no functionality. Obviously, dropping requirements poses an extreme restriction on our product. The product can simply not be complete if not all requirements are adhered to. However, with instruction from our client, we have determined to drop some requirements, implying restrictions on the completeness of our product, to ensure that at least most of the important requirements are taken into consideration. Where time allows, or the project is continued in the near future, the remaining requirements can be implemented.

As explained in the project plan, we have chosen for an incremental approach to implementation and testing. Coinciding with IBM's best practice methods for software design, this implies a modular (but not incremental) design. All functionalities and aspects to be built are to be divided into separate modules and components which will be implemented and tested separately to reduce complexity and increase precision and later integrated coherently resulting in a functional and consistent product, as described in the Requirements Analysis document. To enforce this, the design will seek such separations, ensuring minimal dependencies between the modules to allow for future system evolvement, improvements and adaption.

4.2.1 Goal of Design

The goal of the design is to formally translate all requirements into a complete design which can easily and correctly be translated to code, in order to successfully design and implement an easy to use tool that supports children in their search for information on the Internet.

In order to achieve the main project goal, the previous chapter established a prioritized list of requirements posed by all stakeholders involved in the project. To ensure that the product we are building is the right product, all these requirements must be incorporated in the final product. Zooming in that means that each of the requirements must be described in a design model which in turn can be translated to code. A systematic approach is necessary to guarantee both quality and progress of the project, and to ensure that all goals are met. This document will provide a general product description and identifies the activities to be performed in order to ensure a valid and correct design.

All requirements posed must be translated into a design. To ensure that this is the case, prior to coding we ensure that all requirements have been incorporated in the design by completing a cognitive walkthrough and heuristic evaluation (See section 4.7) during which any requirements which were left out will come to light.

To ensure that the design can be translated into code easily and correctly, we use proven formal design models which break down , describe and simplify the coding process.

Thus, this chapter describes the high-level structure of the software to be built, the responsibilities, relationships, and interactions of components, the relationship between application/technical parts of the system, and specifies how existing, acquired, and developed components are related.

This chapter has been written as a primary form of communication and serves as the unit of work allocation. The encapsulation provided by the components reduces the interaction between the two developers, simplifying the overall management. Because design models are made to be easily interpreted by outsiders, the designs can also be used to communicate with supervisors and client to ensure that expectations are in line with each other.

4.3 High Level Application Model

Looking back at the primary goal of this project, we are creating a system for our users, and their expectations and future use scenarios must form the basis of our design. A system that does not do what the users need it to do will merely not be used. A user model (or use case) can be used to model the functionality and behavior that the user expects from the system. A user model is a graphic representation of the flow of user's effort through the use cases. It describes user activities and expected results, by describing the flow and key process attributes of the product to be made. The flow can be seen as a series of related steps that takes a user input, adds value to it, and yield a (list of retrieval) result(s). Such a model can be refined during the design phase, incrementally zooming in and describing the expected functionality in more and more detail until such a low level depiction has been realized that translation into code has become straightforward and unambiguous.

To inventory the possible functions for which Giggle could be used, we devised such a use case diagram (see figure 4.1) in the initial stages of the Requirements Analysis phase. Feedback on (contents of) the diagram from our user group has contributed to the list of requirements which was the result of the Requirements Analysis phase. In turn, these requirements have been used to refine the model (figure 4.2) to include personalization as an additional functionality (as described in section 3.5.1). For a discussion about and justification of the decisions made thus far, we refer to section 3.

Thus, the aspects depicted in a user model form the fundamental basis of the final characteristics of the product, the user must be able to complete these activities in order for the product to be successful. In order to achieve this, the following two software aspects must be accomplished:

1. **User-enabling functionalities:** The functionalities described in the requirements must be offered in an adequate manner for the user to use, adhering to the users capabilities. The aspects of the interface required in order to make this capable have been described in the Requirements Analysis document and incorporations described in 4.6. Early testing is described in section 4.7.

2. **Supporting functionalities:** The system must be capable of offering the functionalities to the user. The characteristics of these system functionalities are described in sections 4.4 and 4.5. Whether they have been achieved is tested and documented in sections 4.7 and 7.3.

According to the distinction above, we arrange the primary and secondary requirements (from section 3.5.1) into two groups: (1) user-enabling requirements which allow the user to perform specific actions for which we must verify they have been modelled in the use-case model (figure 4.2), and (2) supporting functionality requirements which form the input for the high level analysis model (see section 4.4 in the next section. Categorizing between the two can truncate as a checklist (see table 4.1), simplifying the verification that all requirements have been incorporated in design models and will be, after refinement to lower level specifications, be translated to code.

Requirement	**Category**
Instructions/Help (F1, F3, F9, NF1)	UE
Metaphor (F2, NF1, NF2)	UE
Query input (F4, F5, F6, F9, NF1)	UE
Directory (F3, F4, F5, F9, NF1)	UE
Relationship-map (F4, F5, F9)	UE
Query modification (F5, NF3)	SF
Results list (F7)	UE
Result document (F7)	UE
Reversal of Actions (F8, NF1)	UE
User safety (NF2, NF3, NF4)	SF
Personalization (NF2, NF4)	UE
Links list (NF2)	UE
Time and date (NF2)	UE
Hardware and Software (NF5)	SF
Performance (NF6)	SF
Awareness (B1)	SF
Advanced Search (F9)	UE
Multi-linguistic (B2)	SF
Document Format Types (NF5)	SF
Relevance and Filtering (NF3)	SF
Awareness (B1)	SF

Table 4.1: User-enabling (UE) and supporting-functionality (SF) requirements.

4.3.1 User-Enabling Functionalities

Having distinguished which requirements pertain to the use-case model, we can now validate it. We map each of the user-enabling requirements (categorized in table 4.1) to the scenarios in the use case model (figure 4.2) to ensure that they have all been incorporated in the final model. Because the use case model describes user actions and the requirements describe functions, alternative naming for the same thing has been used. Because this may be confusing to the reader, we show which requirements are fulfilled by which scenario in the user model, including a short description of the action in turns of what the user expects.

- Instructions/Help (F1, F3, F9, NF1)
  Gets Help/Views Introduction: Help on separate page explaining use of all Giggle functionalities, a guided tour, and a sample search. Also includes context sensitive help giving explanation of each specific feature by holding the mouse cursor above the feature;

- Metaphor (F2, NF1, NF2)
  Giggle Mascot: Metaphor to help (including tip-of-the-day) and entertain users;

- Query input (F4, F5, F6, F9, NF1)
  Query Search / Advanced Search: Keyword query input field and 'Find' button to initiate search, including a spell check function which gives suggestions for alternative spellings. It also displays the path chosen by means of selection in the directory or relationship-map;

- Directory (F3, F4, F5, F9, NF1)
  Directory Search: High-level list of categories allowing for selection by mouse which can be iteratively selected to refine search;

- Relationship-map (F4, F5, F9)
  Relationship-map: Relationship-map indicating relationships (synonyms, hypernyms, etc.) between input word (query or category selection) and other words in documents predicted to be relevant.

- Results list (F7)
  Results List: The results list is composed of 5 or 6 documents having obtained the highest relevance score to the query and popular concepts. They are placed in a list ordered by relevance, together with the title and a short summary with query keywords (or synonyms thereof) highlighted and a link to view the document;

- Result document (F7)
  Selects Result: Upon selection of a document, the contents are opened in a new frame. All words in the used query (user query and synonyms) are highlighted, each with a different color. The user is given the possibility to either discard or keep the document, after which information is distilled from the selected document resulting in the reweighting of the active query (and in turn resulting in a new results list);

- Reversal of Actions (F8, NF1)
  Undo/Redo: Back and Forward navigation buttons to undo and redo moves made;

- Personalization (NF2, NF4)

  - *Login:* (Encrypted) Login for existing users and registration for new users by supplying a user name, password, and question-answer combination to recover forgotten password;
  - *Personalizes Site:* Change background colors, toggle 'tip-of-the-day', toggle metaphor, change password, edit list of personal links, edit list of useful predetermined informative links;
  - *Saves Search:* Save search state (user query, reweighted query, results list, relevant docs);
  - *Restores Saved Search:* Load a saved search state (user query, reweighted query, results list, relevant docs).

- Links list (NF2)
  Links List: A predefined list of links to informative sites (dictionary, encyclopedia (encarta.msn.com), atlas, math help (www.purplemath.com), translator (www.worldlingo.com)). Also includes a user defined links list to entertaining sites (games, jokes, comic strips, quizzes, music);

- Time and date (NF2)
  Time and Date: Time and date on main screen.

- Advanced search (F9)
  Advanced Search: Advanced search on separate page with multiple input fields representing different boolean inputs.

The resulting interface from the above list is shown in section 4.6. Whether or not all the functionalities have been incorporated in the design model are tested in section 4.7.

4.3.2 Supporting Functionality Requirements

Of all the requirements, those not listed in the user-enabling requirements are assigned to the list of supporting functionality requirements. As such, we ensure that the lists are both disjunct and complete. The following system functionalities are to be incorporated int the models of sections 4.4 and 4.5 which are in turn to be translated into code. Testing whether the functionalities have been incorporated into the design is tested and documented in sections 4.7 and 7.3.

- Query modification (F5, NF3): Modification of input query using relevance feedback, spell checking input query and query expansion with relevant terms;

- User safety (NF2, NF3, NF4): Improving user safety by filtering undesired aspects from input sites, encrypting user information and warning user of own responsibilities;

- Hardware and Software (NF5): Ensuring system can run on systems available to users and client;

- Performance (NF6): Ensuring the system performs adequately in terms of responsiveness and capacity;

- Awareness (B1): Ensuring that people outside the project knows of its existence;
- Multi-linguistic (B2): Ensuring the system is capable of Dutch, English and Spanish interface and results;
- Document Format Types (NF5): Ensuring multiple document types can be retrieved.

4.4 High Level Analysis Model

The information posed in section 4.3 explained the functionalities and behavior that the final application must exhibit on a high level. However, the description is rather general and cannot directly be used as a recipe for implementation. The intended system behavior is further clarified and specified by means of modelling, in order to allow for an unambiguous specification which can be implemented in order to create the required product. This section describes the first level of the analysis process of the application description. By means of best practice methods such as Unified Modelling and Object-Role Modelling, the application model is translated, step by step, into graphical representations, which can in turn be translated into code for the purpose of implementation. The use of such methods ensure complete, correct and robust translations between different phases in the software process [26][28].

We have chosen to use these methods because they are standardly used by IBM in software development. IBM has had a long history of successful and unsuccessful software developments, yet prides itself on the way they learn from past experiences and new employees. Their experience has led them to establish these methods as their standard, we trust on that. Furthermore, both project members have received education about these methods at the university, ensuring they know how to apply and use the models, are aware of its advantages and disadvantages, and furthermore know that this methodology is appropriate for the problem at hand and has the power to model the system that is to be built.

4.4.1 Modelling Approach

Based on IBM software engineering principles, the following steps indicate how we derived the information represented in our models:

1. Composition of a use-case diagram describing the use-case view and business aspects of the product to be built.
2. An Object Role Model (ORM) Diagram was composed using the use-case view and the remaining requirements described in the Requirements Analysis document (see figure 4.3). This was done by establishing sentences describing use cases and system use cases and then translating these into a model.
3. The ORM diagram was translated into a database description, indicating how to store all required information.
4. Classes and Responsibilities were derived from the ORM using Class Responsibility Collaboration (CRC) modeling.
5. System functionalities were described in natural language by walking through and specifying the different system functionalities (see appendix B.2).

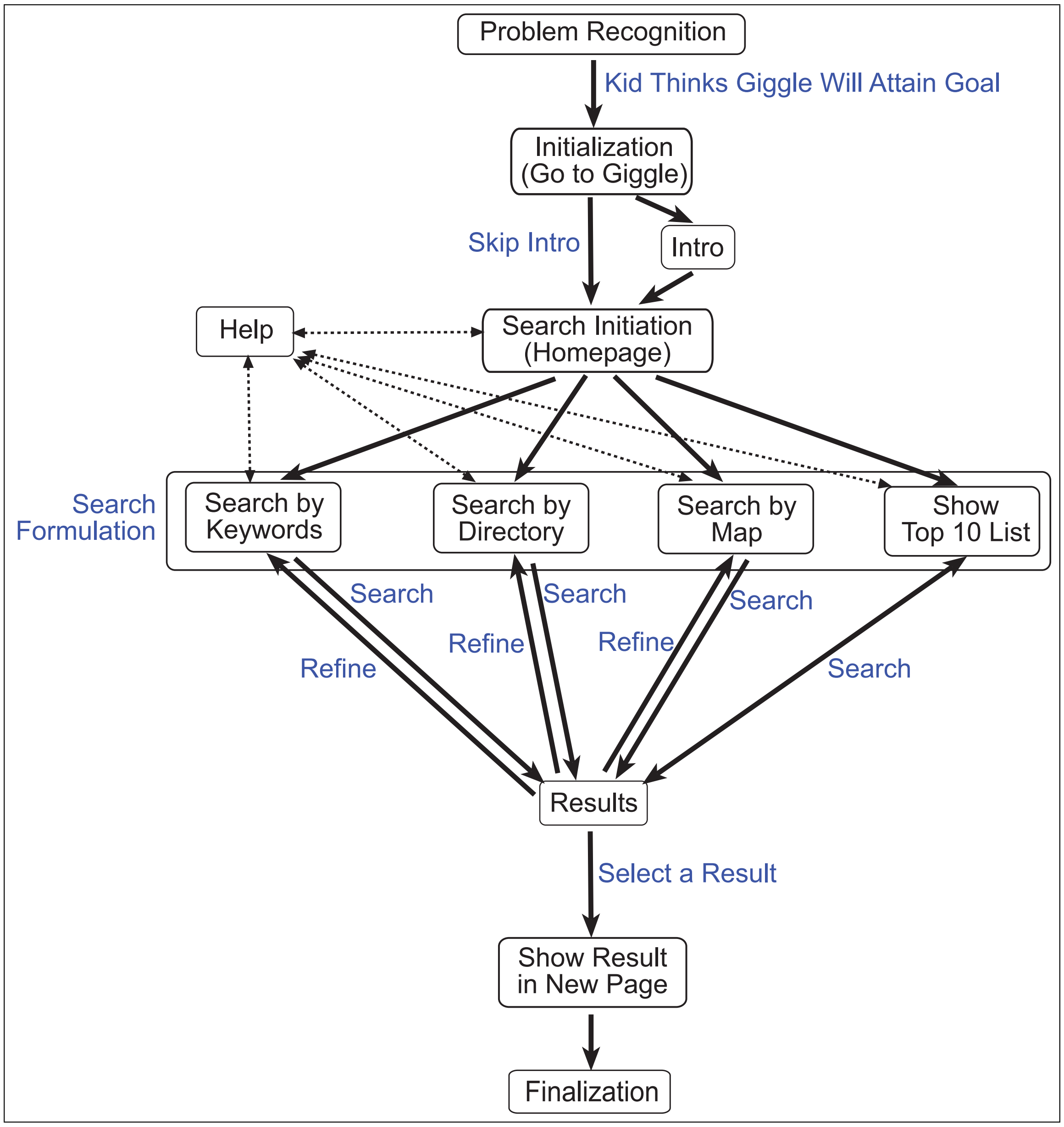


Figure 4.1: Use-Case diagram.

6. Using UML, a class diagram was established showing primary associations that had been described by CRC modelling (See figure 4.5). Arrows represent association relationships, modelling a particular role between two classes. Such a role can be uni- or bi-directional and has multiplicity constraints.

7. The UML use-case view was translated into a logical view by further specifying the design by use of multiple class diagrams, sequence diagrams and state-chart diagrams.

8. The class, sequence and state-chart diagrams were translated to Java code.

4.4.2 Use-case view

Figure 4.2 describes the manner in which our users are to be able to make use of Giggle. Each circle represents a user action. The arrows indicate the direction of flow between actions and subsequent actions.

4.4.3 Object-Role Model

It is well recognized that the quality of a database application depends critically on its design [9]. To help ensure correctness, clarity, adaptability and productivity, information systems are best specified first at the conceptual level, using concepts and language that people can readily understand. Designing a database involves building a formal model of the application area or universe of discourse (UoD). Object-Role Modelling (ORM) simplifies the design process by using natural language, as well as intuitive diagrams which can be populated with examples, and by examining the information in terms of simple or elementary facts. By expressing the model in terms of natural concepts, like objects and roles, it provides a conceptual approach to modelling. Figure 4.3 describes the Conceptual Object Role Model (ORM) diagram. In line with this graphical representation of role models, appendix B.3 gives an overview of the data that must be stored to ensure the correct functionality of Giggle.

4.4.4 Database Description

The ORM that was developed in the previous step first undergoes several steps, namely grouping (collocation of objects with the same primary key, lexicalizing (identifying each column with a unique name), and reduction (removing single column tables with no added value). This normalized relational information-model can directly be translated into a database design.

4.4.5 Class Responsibility Collaboration

This section includes the CRC (Class Responsibility Collaboration) cards that were constructed to indicate the different components of the software to be built, the tasks they must perform and which other components they rely on and thus must be tuned with.

CRC modelling is a best-practice method that ensures that the product to be built is broken down into smaller logical units, ensuring strong cohesion between functionalities within one class, and loose coupling to other classes [26]. This not only dramatically lowers the complexity during further design specification and coding, it is required to ensure high flexibility, adapting one class requires minimal adaption of other classes. Furthermore, this description can be used at a later stage to determine which other classes will be involved when one is

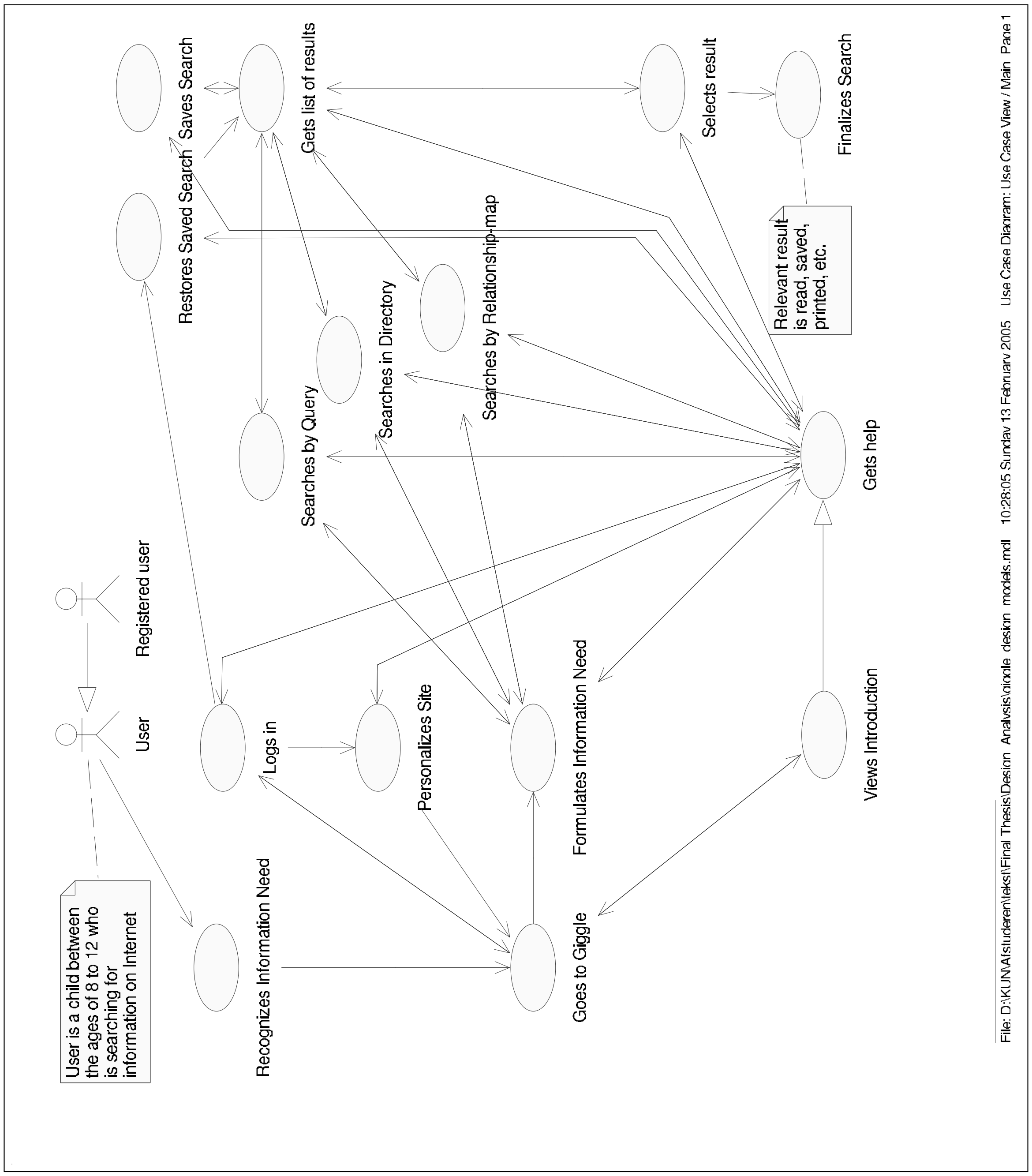

User
Registered user
User is a child between the ages of 8 to 12 who is searching for information on Internet
Recognizes Information Need
Logs in
Personalizes Site
Goes to Giggle
Formulates Information Need
Views Introduction
Gets help
Searches by Query
Searches in Directory
Searches by Relationship-map
Restores Saved Search
Saves Search
Gets list of results
Selects result
Finalizes Search
Relevant result is read, saved, printed, etc.
File: D:\KUN\Afstuderen\tekst\Final Thesis\Design Analysis\giggle design models.mdl 10:28:05 Sunday 13 February 2005 Use Case Diagram: Use Case View / Main Page 1

Figure 4.2: Use-Case diagram.

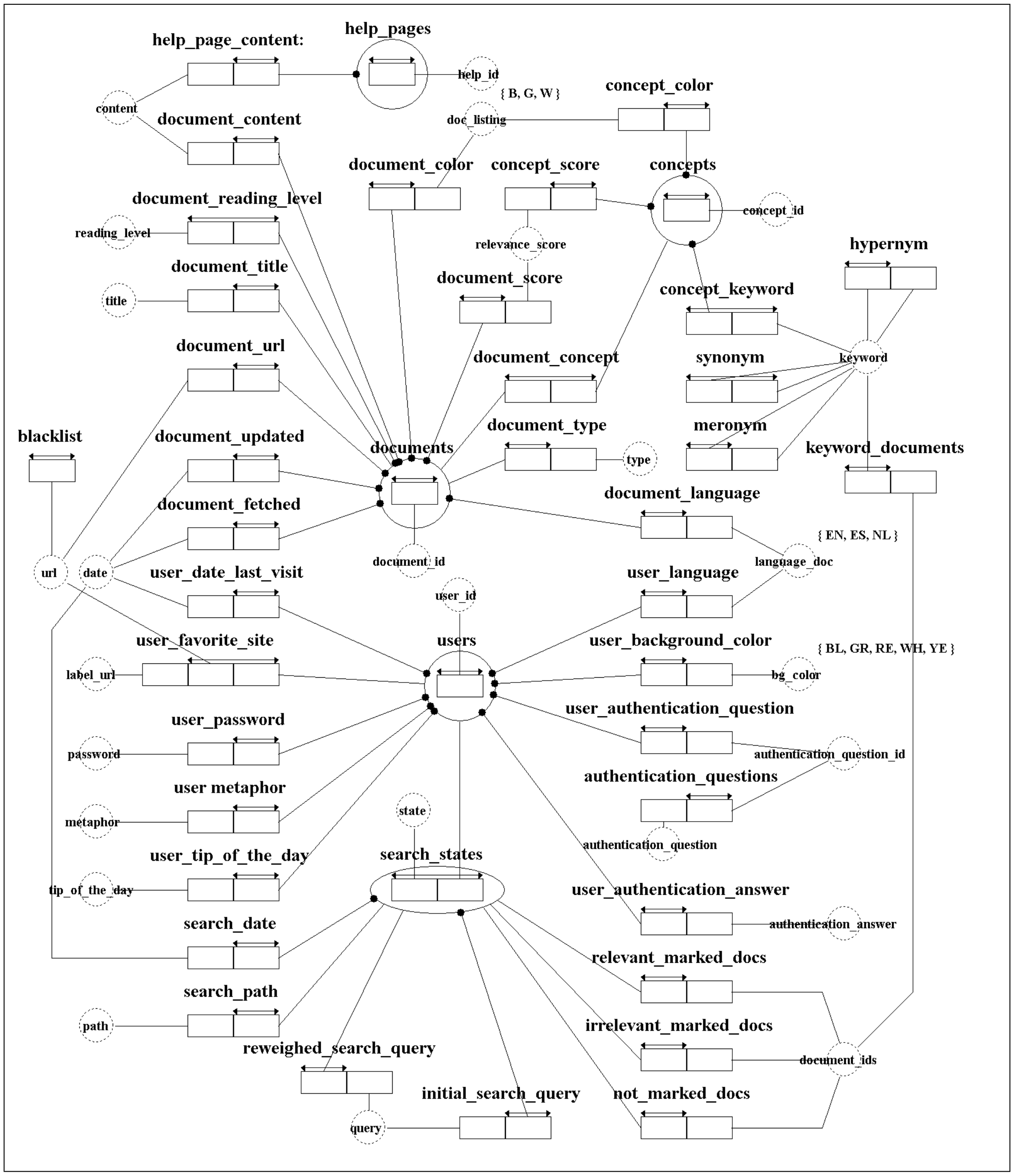


Figure 4.3: Conceptual Object Role Model (ORM) for database creation.

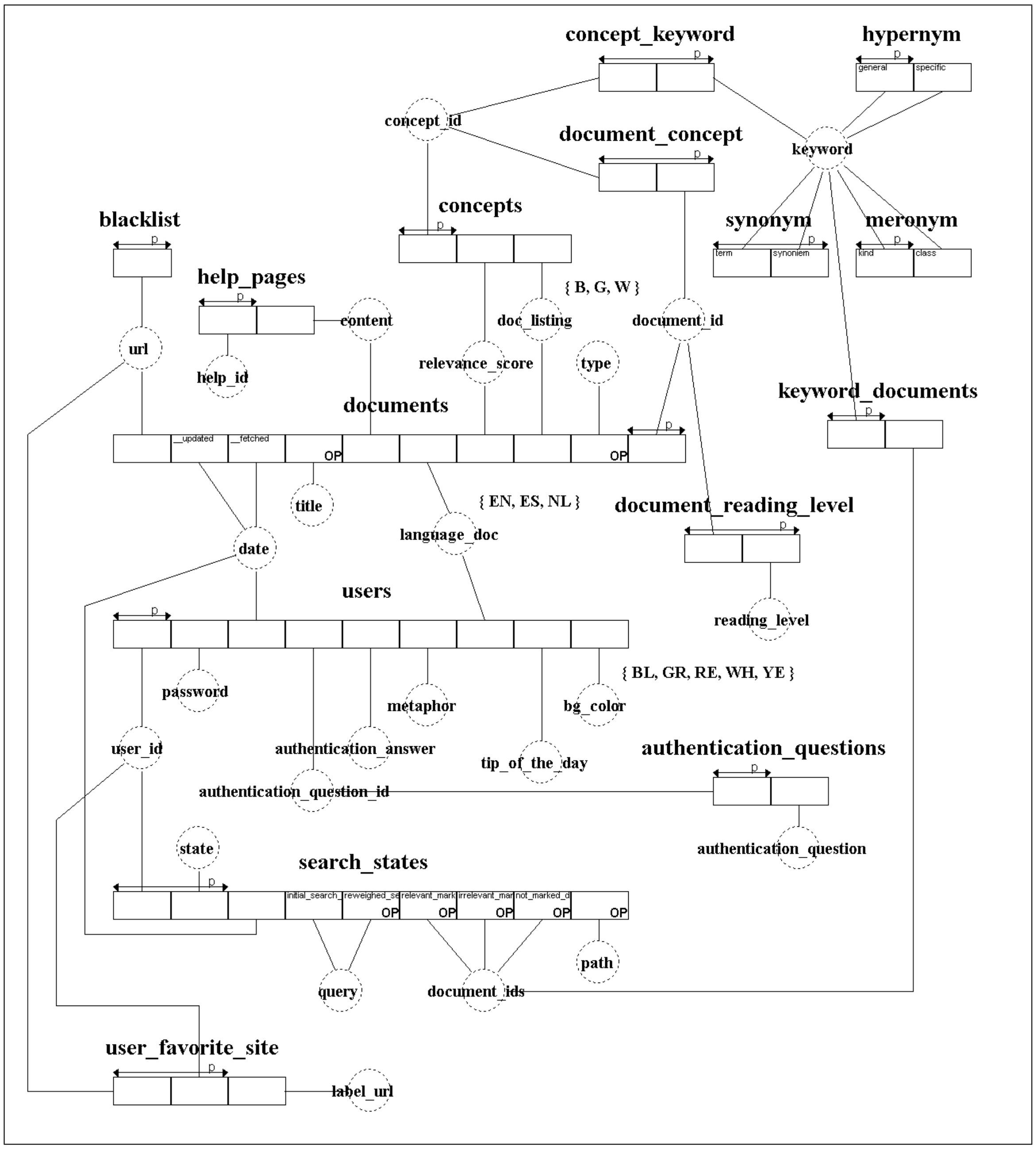


Figure 4.4: Normalized Object-Role Model.

adapted or the system evolves.

Class:	**Active_Search**
Cluster:	User Personalization
Responsibility:	Stores user query Rewieghts user query Retrieves Documents Determines document relevance score
Collaboration:	Query_Input, Directory, Relationship_Map
Subclasses:	Results_List

Class:	**Analyzer**
Cluster:	Retrieval Functionalities
Responsibility:	Analyzes document content to determine content topic Determines doc+query vectors Determines concepts Reweighs vectors (user relevance feedback) Analyzes concepts prominent children docs
Collaboration:	Database, Retrieved_Doc, Bright
Subclasses:	User_Relevance_Feedback

Class:	**Database**
Cluster:	Database
Responsibility:	Stores and retrieves information
Collaboration:	Analyzer, Resulting_Doc
Subclasses:	Blacklist, Concept_list, Word_lists, Whitelist, User_help, User_info

Class:	**Directory**
Cluster:	Interface
Responsibility:	Shows directory categories on screen Allows for iterative search
Collaboration:	Results_List, Search_State

Class:	**Document_Preprocessing**
Cluster:	Document Preparation
Responsibility:	Removing accents, spacing, punctuation, capitalization Removing stopwords determining noun-group pairs Stemming
Collaboration:	Tagger, Lexicon, Retrieved_Doc

Class:	**Lexicon**
Cluster:	Document Preparation, Retrieval Functionalities
Responsibility:	Determines language of a document Determines reading level of a document Constructs word lists of synonyms and related words Query expansion determines document reading level
Collaboration:	Resulting_Doc
Subclasses:	Language_Determiner, Reading_Level_Determiner, WordNet tables

Class:	**Login**
Cluster:	User Personalization, Interface
Responsibility:	Allows user to login Allows user to register Recovers lost password
Collaboration:	Registered_User, Database

Class:	**Police**
Cluster:	Retrieval Functionalities
Responsibility:	Checks retrieved documents for appropriateness Determines document listing Sets blacklisting for storage
Collaboration:	Resulting_Doc, Database

Class:	**Query_input**
Cluster:	Retrieval Functionalities
Responsibility:	Reads query input from screen Posts modified query to screen
Collaboration:	Active_Search_State, Lexicon

Class:	**Reading_Level_Determiner**
Cluster:	Document Preparation
Responsibility:	Determines reading level of a document
Collaboration:	Resulting_Doc
Superclass:	Lexicon

Class:	**Registered_User**
Cluster:	User Personalization, Interface
Responsibility:	Registration/Login Stores/Loads search
Collaboration:	Active_Search, Login
Superclass:	User

Class:	**Relationship_Map**
Cluster:	Interface
Responsibility:	Shows word relationships in a 3D-map on screen Allows for iterative search
Collaboration:	Results_List, Search_State, Database, Query_Input

Class:	**Resulting_Doc**
Cluster:	Retrieval Functionalities, Interface
Responsibility:	Show content Allow relevance marking Set relevance marking
Collaboration:	Database

Class:	**Results_List**
Cluster:	Retrieval Functionalities, Interface
Responsibility:	Maintains a list of relevant documents Posts list to screen
Collaboration:	Resulting_Doc, Active_Search_State

Class:	**Spider**
Cluster:	Database
Responsibility:	Crawls the web in search of documents Makes docs ready for use by our system
Collaboration:	Resulting_Doc, Tagger

Class:	**Tagger**
Cluster:	Document Preparation
Responsibility:	Tags documents
Collaboration:	Resulting_Doc, Spider

Class:	**User_Relevance_Feedback**
Cluster:	Retrieval Functionalities
Responsibility:	Adapts query according to user document preferences
Collaboration:	Active_Search, Results_List, Resulting_Doc
Superclass:	Analyzer

4.4.6 Unified Modelling Language Model

The Unified Modelling Language (UML) is a general-purpose visual modelling language that is designed to specify, visualize, construct and document the artifacts of a software system. It is used as a best practice method by IBM. We have previously established that we are confident in the quality of the engineering methods that IBM uses, and thus are also confident in the quality of UML as a software engineering method. The UML meta-model is defined as one of the layers of a four-layer meta-modelling architecture. This architecture is a proven infrastructure for defining the precise semantics required by complex models. The benefits of adapting the formal UML techniques include the improvement of correctness of the description, reduction of ambiguities and inconsistencies, validation of the architecture of the meta-model, and increased readability of the description.

Class Diagram

Description

The Class Diagram is a structural representation of the software objects and their static relationships that comprise the system being developed, using an object oriented approach. This description is made using the following structuring concepts, and the model includes a detailed description of each:

– Classes (including instances of classes)
– Attributes (representing the knowledge responsibilities or data)
– Methods (representing operational responsibilities or functions)
– Association relationships between classes
– Aggregation relationships between aggregate and component classes
– Inheritance relationships between super-classes and subclasses
– Formal or informal constraint descriptions (optional)

Purpose

The overall purpose of the Class Diagram is to interpret business, user and system requirements and develop an overall model of what is expected of the software. Creating a robust and complete Class Diagram further helps to accomplish the following goals:

- Refine the system boundary or context by deciding which objects are within the boundary of the software system;
- Create a starting point for other architectural layers such as the user interface and the database model;
- Help divide and organize work by partitioning the problem domain into separate "areas of concern" based on the relevant objects, this allows us to partition the application in the best way to support parallel development efforts;
- Facilitate the verification and validation of the analysis and design by enabling a high-level representation of requirements in a detailed and concise manner
- Avoid rework in programming by considering changes and realizing the key abstractions, patterns, and mechanisms prior to implementation;
- Enhance the quality of the design by facilitating the building of design structures for commonality and malleability and encourage common approaches to the use of common design principles, architectural templates and design patterns;
- Improve the understanding of the design and system intent by the development team and key customer personnel to allow the actual construction to be done both iteratively and incrementally.

Results
The Class Diagram is typically developed iteratively over the life-cycle of the project. Several different iteration stages can be depicted, each specifying elaboration points during development:

1. **Conceptual View:**
   The initial or conceptual view (see figure 4.2) focuses on what is traditionally thought of as analysis, i.e., "what" is needed for the solution. Analysis is concerned with defining the problem domain by understanding what aspects of the user model are to be included in the software system. As the decisions about how the software system will be constructed are not the primary concerns at this time, the design remains technology neutral, although not technology ignorant.
2. **Logical View:**
   The logical design or specification view is focussed at searching for answers to the questions on "how" the system will be implemented and where the overall structure of the solution is defined. Figure 4.5 shows an overview of the classes and their corresponding associations. For abstraction purposes, packages were created. Packages group classes that show strong cohesion with each other and loose coupling with classes in other packages. Figure 4.7 gives a more specific indication of the generalizations, associations and dependencies in the most complex package, the search package. For purpose of clarity and maintaining overview, subclasses have not been modelled, rather, in the model each superclass is appointed the responsibilities of its subclasses; the precise boundaries in responsibilities and functions have been described in section 4.4.5. Factors such as concurrency and distribution,

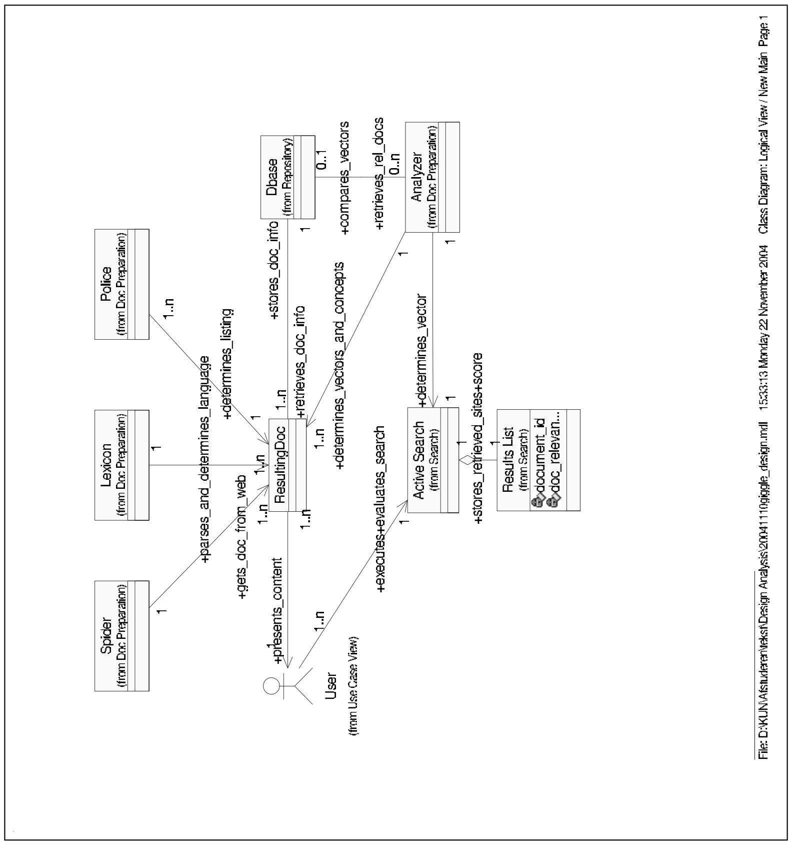


Figure 4.5: Logical View: main class diagram.

coordination and sharing, transactions and persistence, user interface capability, and system interfaces such as communication are also taken into account as well. During this stage in the design process, much of the design is dependent on the technology and architectural decisions have also been made at this time.

3. **Component View:**
   The final iteration is the physical design or implementation view that details the constructs and mechanisms to be used based on the actual implementation language chosen, in this case Java. This stage may include the reorganizing existing classes and the introduction of new classes to handle implementation specific details.

Interaction Diagram

Description

An Interaction (or sequence) Diagram is a graphical representation of what depicts the internal behavior of the system. It shows how objects collaborate by sending messages and returning responses to each other. Typically, the initial diagrams only show the key interactions between the main user-related objects discovered during analysis of

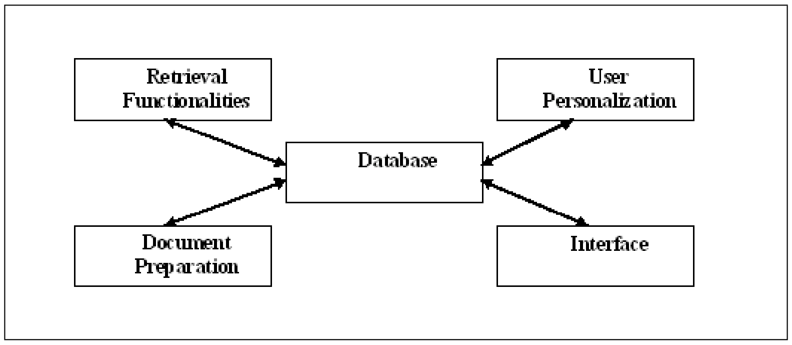


Figure 4.6: Logical View: high level class diagram.

use cases. Later iterations will also include invented objects that provide the control, interface, communication, distribution and storage functionality of the system.

An Interaction Diagram presents the dynamics of a scenario by representing related objects, messages, and interaction flows. It shows how objects can collaborate to perform a scenario. They reflect how the system manages interface, communication, control, and persistency based on the system requirements and the chosen architecture.

Purpose
Interaction Diagrams form a direct link between the functionality specified in the use cases and the functionality that can be supported by the Class Diagram. It's main intent is the for allocating behavior to objects, discovering problems, holding design discussions and considering alternatives to the current design. Interaction Diagrams can be used to help identify classes and their responsibilities by requiring the following decision to be made:

– Which objects of which classes participate in the Interaction Diagram and what their behaviors and responsibilities are;
– An ordering of the operations relevant to carrying out each particular scenario;
– An allocation of the responsibilities for these operations to specific classes.

Interaction Diagrams provide an end-to-end understanding of how objects can carry out the scenario, which is difficult to achieve from the study of a Class Diagram alone.

Result
Figures 4.8, 4.9, and 4.10 are sequence diagrams, each indicating involvement of classes per scenario, the functions required to complete the task, and the sequence in which the functions are to be called.

Statechart Diagram

Description
A State-chart Diagram represents the behavior or life-cycle of a single class; it is a projection of all Interaction Diagrams for a single class. It does this by describing:

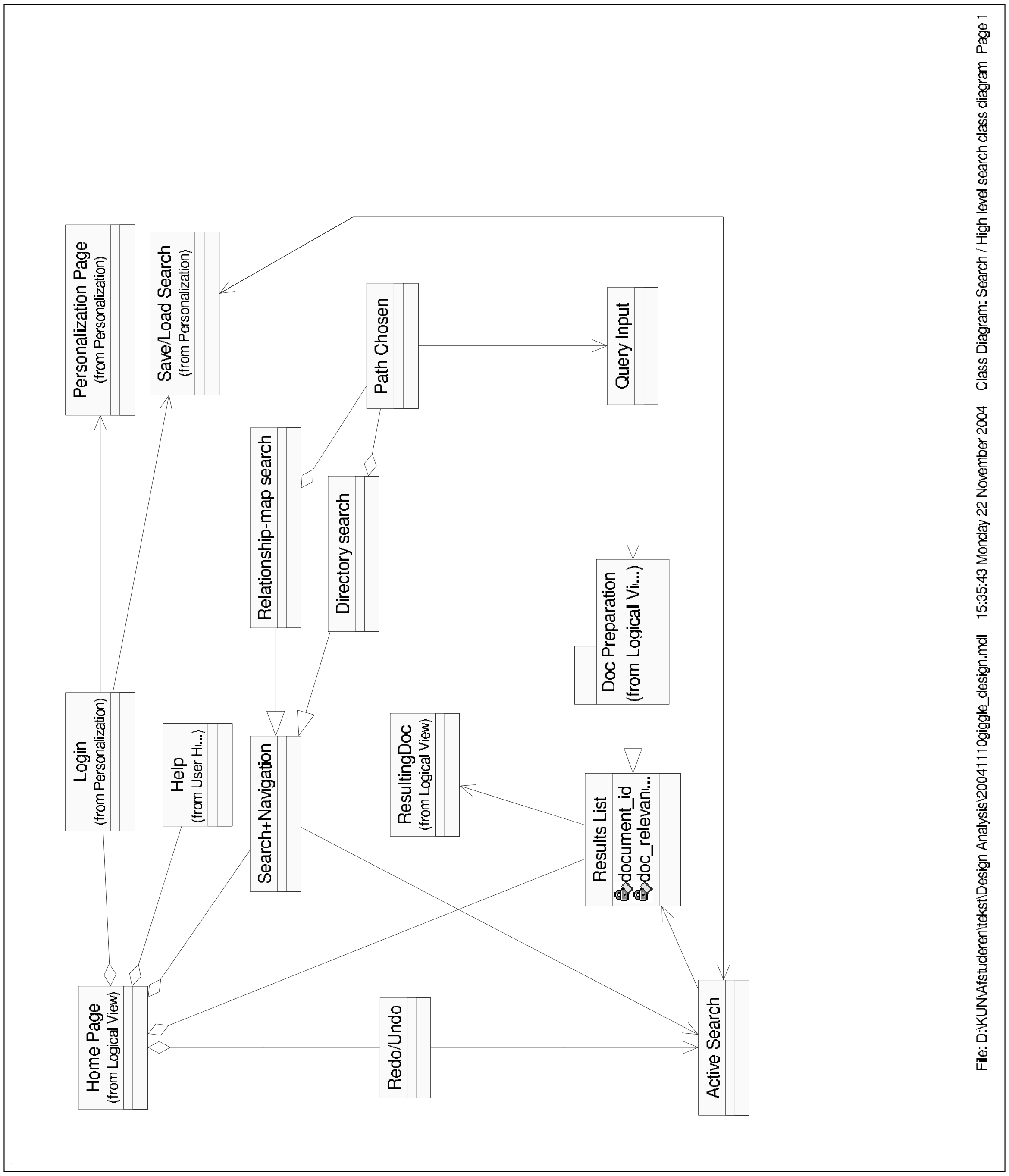


Figure 4.7: Logical View: high level search class diagram.

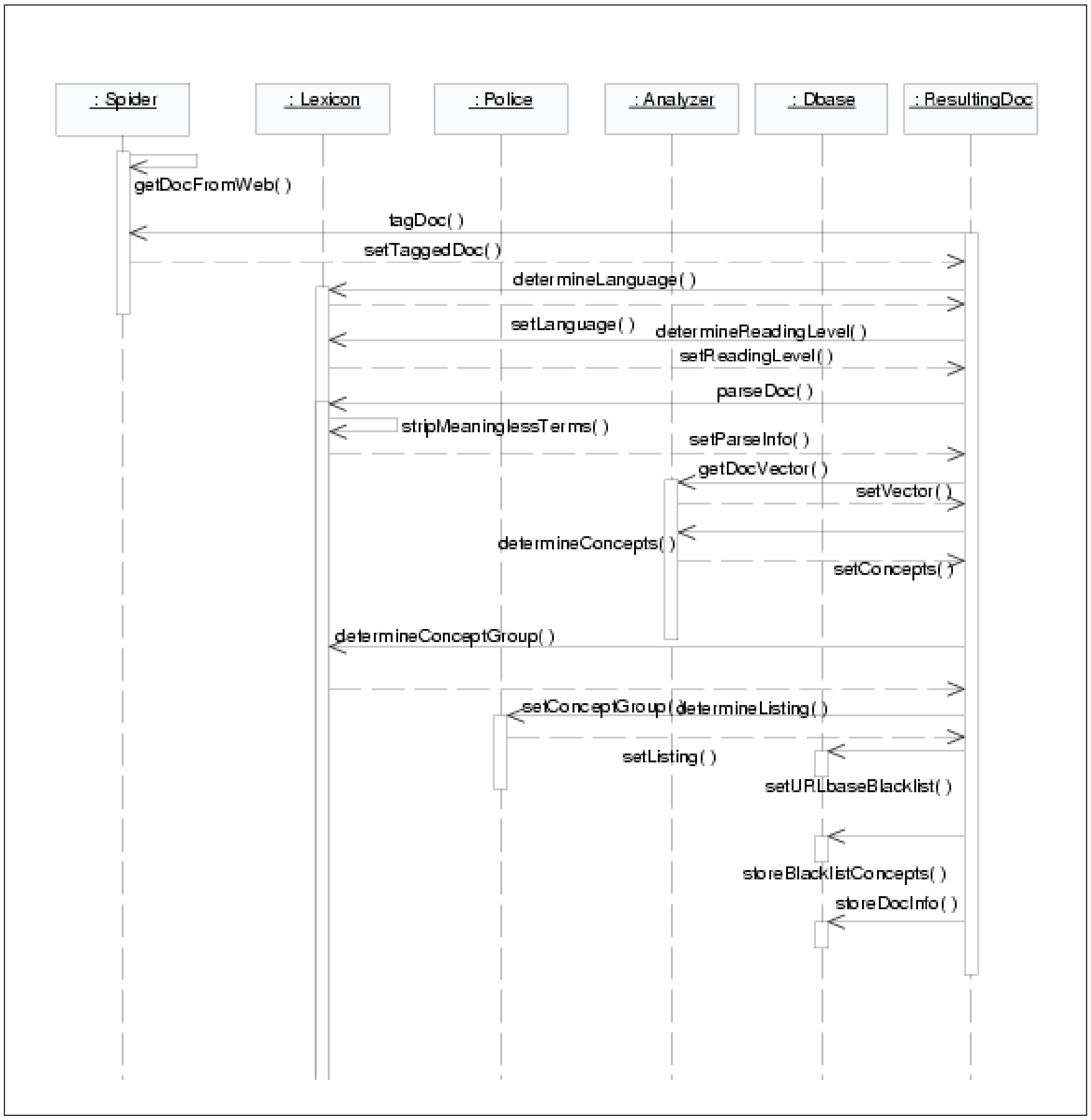


Figure 4.8: Sequence Diagram: Document Preparations.

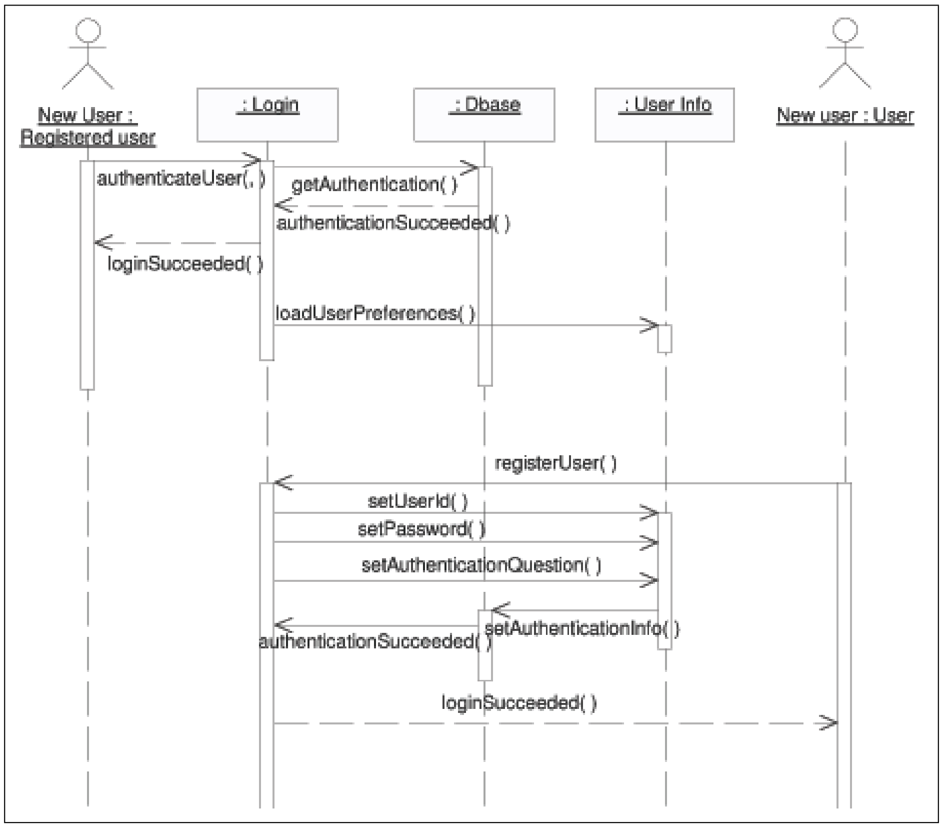


Figure 4.9: Sequence Diagram: Registration.

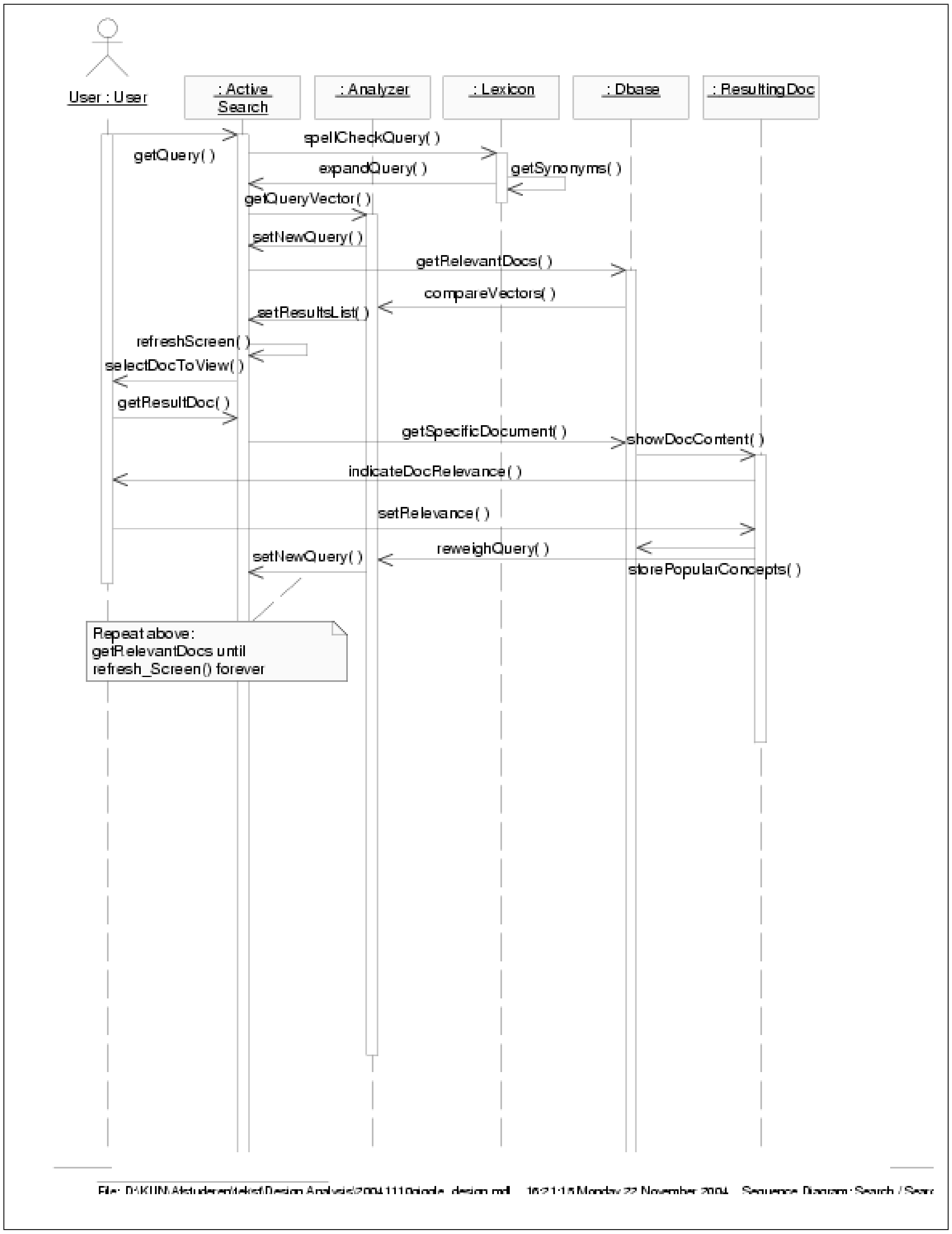


Figure 4.10: Sequence Diagram: Query Search.

- States that the may be attained;
- How receipt of events, messages and responses cause a change in state and behavior.

Purpose
The State-chart Diagram provides a convenient, visual means of understanding how the threads of interaction among objects, as shown in Interaction Diagrams, can be projected onto each class participating in these interactions. State-chart Diagrams created during Analysis are done for the purpose of discovery or understanding. Only classes with strong state-dependent object behaviors need to have a State-chart Diagram, one should not attempt to model trivial behavior using a State-chart Diagram. It is for this reason we have chosen only to model the state of the repository class during document preparation. A graphical representation helps improve the ease of reading and understanding. The event that triggers the interaction/transition, under which condition it is triggered, and what actions are accomplished.

Result
Figure 4.11 is a State-chart describing which actions must be taken under which circumstances during document preparation for storage in the repository.

4.5 High Level Architecture Model

4.5.1 Architecture Overview

Description

The Architecture Overview diagram (see figure 4.12) is a schematic diagram that represents the governing ideas and candidate building blocks of an IT system to be built. It provides an overview of the main conceptual elements and relationships in an architecture, which frequently include candidate subsystems, components, nodes, connections, data stores, users and external systems. Its main purpose is communication.

Purpose

The Architecture Overview is used to:

- Communicate to the client and external stakeholders a conceptual understanding of the intended IT system
- Provide a high-level shared vision of the architecture and scope of the proposed IT system for the entire development team
- Explore and evaluate alternative architectural options

Key Concepts

- The architecture describes how the different intermediate components communicate with each other on a high level. The left hand side of figure 4.12 shows how the users communicate with the system through the interface. The bottom of the figure shows how the system retrieves it's information from the web, through the indexer.

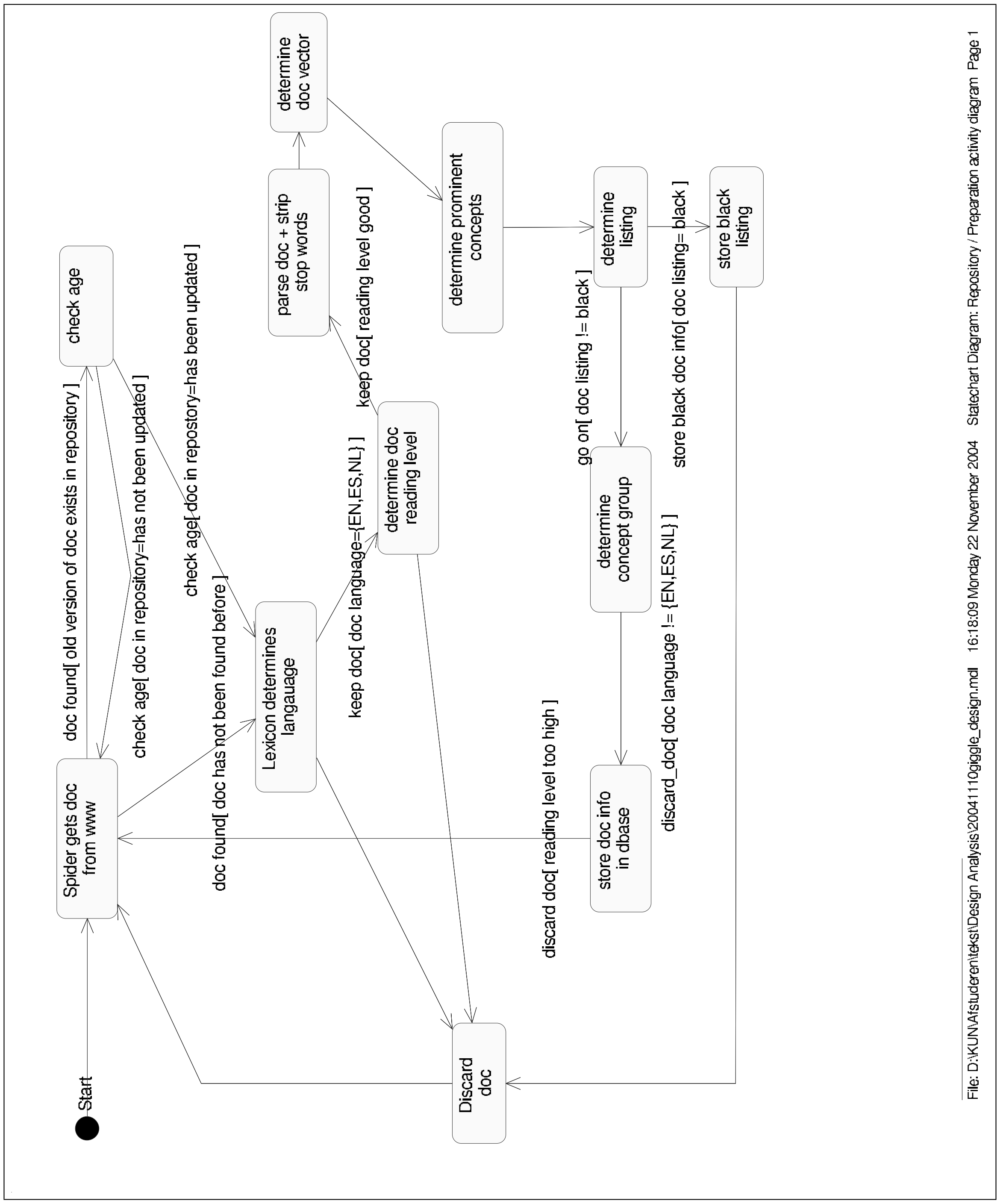

Start
Spider gets doc from www
check age
doc found[old version of doc exists in repository]
check age[doc in repository=has not been updated]
check age[doc in repostory=has been updated]
doc found[doc has not been found before]
Lexicon determines langauage
keep doc[doc language={EN,ES,NL}]
determine doc reading level
keep doc[reading level good]
parse doc + strip stop words
determine doc vector
determine prominent concepts
determine listing
go on[doc listing != black]
store black doc info[doc listing= black]
store black listing
determine concept group
discard_doc[doc language != {EN,ES,NL}]
store doc info in dbase
discard doc[reading level too high]
Discard doc
File: D:\KUN\Afstuderen\tekst\Design Analysis\20041110giggle_design.mdl   16:18:09 Monday 22 November 2004   Statechart Diagram: Repository / Preparation activity diagram Page 1

Figure 4.11: State Chart Diagram: Repository Preparation.

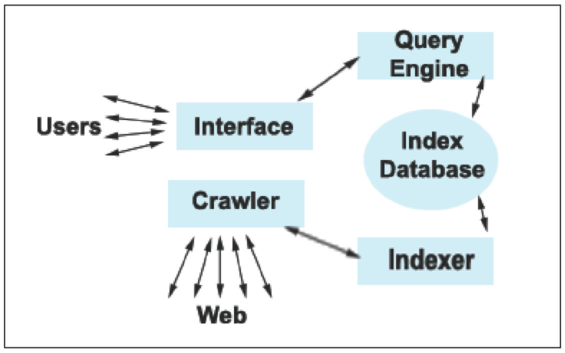


Figure 4.12: Architecture overview.

- To ensure that the number of communication-lines are restricted, there is a separation between functions that retrieve information from the Internet or database from other functions. As such, only one class component can communicate with the database, and only the indexer can retrieve information from the Internet.
- The proposed architecture is a three-tier system, with client workstations (the user computer), a webserver (to host the site) and a database server (repository to store document and user information).
- The proposed system provides access to several remaining components, including User Data, Document Data, and Lexical Data.

4.5.2 Component Model

Description

The Component Model work product describes the entire hierarchy of components in terms of to their responsibilities, their interfaces, their (static) relationships, and the way they collaborate to deliver required functionality. A component is a relatively independent part of a system. It is characterized by its responsibilities and eventually by the interface(s) it offers. A component can be a software subsystem, a program module, a collection of classes, a program, a part of a product, or a hardware device.

Purpose

The Component Model is used to:

- Describe the high-level structure of the software.
- Describe precisely the responsibilities, relationships, and interactions of components.
- Document how application/technical parts of the system are related.
- Specify how existing, acquired, and developed components are related.
- Reduce complexity through the encapsulation offered by a component.

- Serve as the unit of work allocation. The encapsulation provided by the components reduces the interaction between the two developers, simplifying the overall management.
- Define precisely the boundaries between reusable parts of the system.

Component Description

This section describes which individual components can be distinguished. The interaction between these components has been indicated in figure 4.5. A specification of each component (in alphabetical order by component name) follows below:

- **Active Search State**
  Responsible for maintaining information about the current search state (e.g. user query, reweighted system query, path of selections made in directory/relationship-map, documents marked irrelevant, documents marked relevant, date of search, etc.).
- **Analyzer**
  The Analyzer is responsible for analyzing queries and documents and translating these to vector form. It is responsible for reweighting queries where appropriate (in line with user relevance feedback, popular concepts, or stop word removal). It is also responsible for comparing vectors to indicate relevance.
- **Database**
  The Database is responsible for storing all information. It must be capable of handling SQL statements for updates, selection, and insertions. We have chosen for an off-the-shelf product DB2, because it has been made readily available to us from IBM and furthermore is adequate for the uses we expect.
- **Directory**
  The Directory component is responsible for building a directory and adapting accordingly to user selections. The Directory is static and is predetermined by hand. Goal is to make high level, rather disjunct, categories. Approach is by using standard library categories and those of popular childrens search engines as a basis, and gathering documents (favorable to children, possibly with known popular concepts), analyzing content and determining relevant categories. It is permitted to place a document in more than one category.
- **Document Preprocessing**
  The Document Preparation component responsible for preparing documents for retrieval purposes, stripping (removing accents, spacing, punctuation, and capitalization), removing stopwords, determining noun-group pairs, and apply stemming.
- **Help**
  The Help is responsible for supporting the user when he runs into problems. It includes a context-sensitive help explaining every function on screen in short, a guided-tour giving a search example, and a more elaborate explanation of each function separately.
- **Interface**
  The Interface is responsible for showing up-to-date information to the user in an appropriate and timely fashion. Aspects are the results list, relationship-map, directory, path

chosen by user, query input field and search button, alternative spellings, tip-of-the-day, links, help/search tips, forward/backwards, metaphor, login/registration/logout functionality, and all preference aspects pertaining to a personalized interface.

- **Language Determiner**
  The Language Determiner component is responsible for determining if a document has been written in English, Dutch, Spanish or neither.

- **Lexicon**
  The Lexicon component is responsible for finding relevant words and concepts to a given term. This includes composition of word lists of synonyms, meronyms, hypernyms and hypernyms. It must also maintain lists of concept-keyword relationships.

- **Login**
  The Login component is responsible for the login of the user. It includes encrypted login (corresponding user id and password), registration of a new user, renewal of password, authentication by question-answer method, masking passwords to requirements posed on them (8 characters of which 2 are non-alphabetical). All user information must be encrypted during transfer.

- **Police**
  This component is responsible for determining if a tagged document belongs to a black-list or not. It does so by inspecting the words in the document, any blacklist words makes the entire document blacklisted. The URL-base of each link in a blacklisted document must be stored in a url-base blacklist.

- **Query Input**
  Responsible for retrieving user query from screen and posting modified (yet legible) query to screen.

- **Reading-Level Determiner**
  This component is responsible for determining the reading age of a tagged document according to existing algorithms.

- **Relationship-Map**
  This component is responsible for presenting word relationships in a 3D-relationship-map given a base keyword. Assuming we have in the database a set of known popular concepts (initially filled by retrieving popular documents from childrens search engines), the approach is to use WordNet to pinpoint terms related to those concepts. Concept groups defined by a conceptual model can be used to determine alternatives. The original concepts and the related words are placed in a 3D map; empirical distance is used to represent distance in synonymity, and physical location (above, below or besides) and colors are used to represent hypernyms, meronyms, and hyponyms.

- **Resulting Document**
  This component is responsible for showing the resulting document in adequate fashion to the user. Upon selection of a document from the results list, the document contents are to be shown in a new frame. All words in the used query (user query and synonyms) are highlighted, each with a different color. Other relevant document information (e.g. URL) is also to be shown. Furthermore, The user is given the possibility to either discard

or keep the document (indicating relevance). After relevance indication, information is distilled from the selected document and used to reweight the active query (and in turn resulting in a new results list). If relevance is indicated, popular concepts found in the document are added in the database.

- **Results List**
  After user input, the Results List is responsible for indicating the most relevant documents found pertaining to the query. These documents are to be arranged in order of ranking relevance, a short passage with query terms (user input terms and synonyms) highlighted. Preferably, the document's best passage should be shown, a window of text that is most relevant to the user's query providing a better indication of the document's relevance [108], rather than using the first few words to summarize the document on a results page. If XML standard has been adopted, this can be used for a reliable summary [108]. The list furthermore has a link which can be used to view the resulting document in a new frame. It must also show if the user has already indicated that a particular document is relevant. Documents marked as irrelevant by the user should be removed from the results list.

- **Spell Check**
  This component retrieves possible alternative spellings based on the edit-distance algorithm, applicable to terms not found in the dictionary.

- **Spider**
  The Crawler is responsible for retrieving documents from the Internet and placing them, and any relevant information (URL, date retrieved, date last updated, etc.) in the database.

- **Stemmer**
  The Stemmer is responsible for returning stemmed variants (grammatical roots) of terms according to the Porter algorithm.

- **Tagger**
  The tagger is responsible for preparing documents retrieved by the crawler for further use. This includes the stripping of HTML (and other) codes, and placing a document identification tag in front of each sentence in the document. It must thus be capable of determining where each sentence initiates in a document.

- **User Relevance Feedback**
  This component is responsible for dealing with the user relevance feedback. Upon indication of relevance of a document, this component is to ensure that appropriate reweighting of the query takes place, remove documents marked irrelevant from screen and mark documents marked relevant on the screen.

4.5.3 Architecture Decisions

Description

This section documents important decisions about all aspects of the architecture including the structure of the system, the provision and allocation of function, the contextual fitness of the system and adherence to standards.

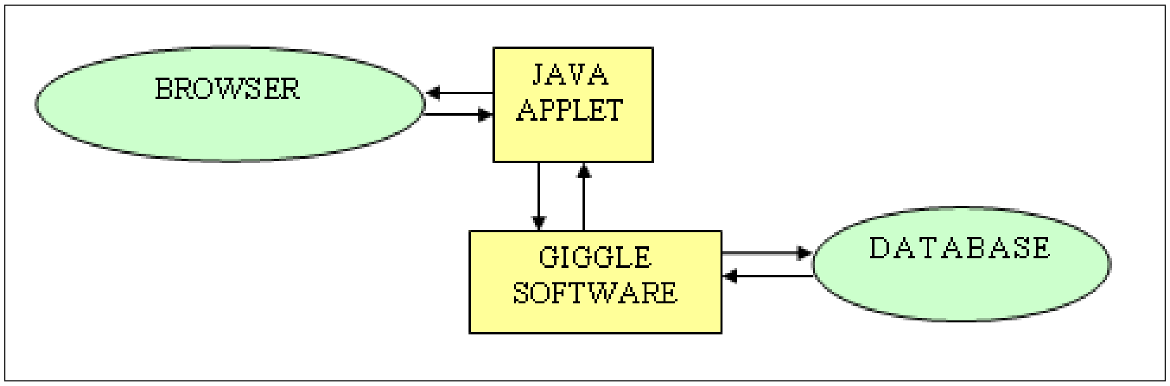


Figure 4.13: Client/Server Architecture

Purpose

The purpose of the Architectural Decisions work product is to:

- Justify architectural decisions;
- Provide a single place to find important architectural decisions;
- Preserve design integrity in the provision of functionality and its allocation to system components;
- Ensure that the architecture is extensible and can support an evolving system.

Architecture Description

The basic WWW architecture, on which Giggle is constructed, is based on the client/server model of distributed systems, depicted in figure 4.13. In the model, a client process makes a request to a server process, normally running on a different machine and using a network such as the Internet for communication. The server process receives the request, establishes a connection with the client, performs the desired function, returns the result to the client, and breaks the connection. The server is then available to receive requests from other clients and perform similar services for them. As can be seen in figure 4.12, the entire system is made up of the following components, which from a higher level architecture, either belong to the server-side or the client-side of the system:

- *Interface*: The interface is made using Java Applets. These themselves have two components, namely Java code which describes the behavior of the applet, and HTML code which calls the applet from a browser (usually Internet Explorer or Netscape). As such, the Java Applet reside on servers and are downloaded and run on the client using a browser.
- *Query Engine*: A request from the user (for example a query input) is sent from the client to the server. As effect, a program is executed which accesses the database in order to generate the desired results and communicate these back to the client. The query engine is thus part of the server-side of the system.

- *Index Database*: Data repository located on the server-side.
- *Crawler*: A program execute from the server-side of the system that traverses the internet and posts requests to websites to retrieve their contents. These are handed to the indexer for preprocessing.
- *Indexer*: A program running on the server-side, that given website contents analyzes them and stores relevant information in the database repository.
- *Web*: An existing medium (neither client nor server side) which enables communication by means of standard protocols.

Database Tools

As a data repository, we have chosen to use IBM's database management system (DBMS) named DB2. DB2 stands for Database 2, a family of relational database products offered by IBM. DB2 provides an open database environment that runs on a wide variety of computing platforms. DB2 offers many advantages with respect to maintenance and retrieval of large amounts of data, including fast operations on data, compact storage of data on disk, and reduction of data redundancy by data sharing. Prominent reasons for choosing this database is availability of the tool and experts at IBM that gave us a crash course on how to use DB2, gave us advice about pitfalls, uses and performance issues, and furthermore were available for troubleshooting and help us if we ran into problems. Furthermore, because this tool is to be used at IBM in the future, making use of their own tools seems intuitive.

Because DB2 can be accessed by standard SQL, the barrier to learn it is low (both programmers have extensive knowledge of SQL). This also makes it easy to interchange data between DB2 and other databases. Furthermore, because a DB2 database can grow from a small single-user application to a large multi-user system, there is room for the application to grow immensely in the future. An example is in the consideration of parallel processing, a feature prominent in any large search engine.

Another advantage of DB2 is that it allows for automatic maintenance to improve performance. To speed up the process of retrieving the right information from large tables, databases use indexing, which must be indicated in advance. DB2, on the other hand, uses statistics to determine whether indexing or table scanning is the fastest for row retrieval from a table (which is generally dependent on the size of the table). Besides, DB2 has also the capability of performing reverse scans (initiating a search at both the bottom and the top of a table) increasing speed without deteriorating performance, and it automatically efficiently stores data.

Programming Tools

Several different programming languages lay at hand for the implementation of our tool. The design guidelines (section 3.5.2) distilled from IBM's best practices indicate that Object-Oriented programming is most flexible, adaptive, dynamic, and allows for reusability. It also adheres to our manner of modular design, enabling us to make full use of our design models and simple translation to code. It also simplifies testing because the classes represent the same actors as described in the design models. The language must also be capable of communicating with a web-application. Due to the time restriction (we did not have sufficient time to learn a complete new language) we had to choose a language we both felt comfortable with. As a result we decided to program in Java.

Our decision to use Java in combination with Java Applets was based on adequacy for the product goals (capabilities), compatibility with other systems (DB2 and web) availability of tools (shareware), tutorials (wide-spread popularity), and our own previous knowledge, experience and expertise. As both members of the team were experienced with both Java and object oriented programming, this was advantageous.

The Java programming language was developed at Sun Microsystems Inc. Java is a platform-independent, object-oriented language. A Java applet is the result of running a Java compiler on Java source code. Applets reside on servers and are downloaded to clients when referenced via a URL. Applets execute on remote sites under the guidance of the browser that downloaded them. Java is designed to be secure and robust enough to prevent any applet's byte code's instructions to compromise the client machine's system.

Also, as explained in the previous section, DB2 in combination with SQL will be used for reasons of uniformity with IBM and existing extensive knowledge of the programmers.

Furthermore, because we are working in a project team and need to be able to exchange our files, from and to different locations, as well as maintain a list of the newest versions, we have chose to use a CVS system. CVS (Concurrent Versions System) is a version control system. It is useful for everyone from individual developers to large, distributed teams. It lets developers access the latest code from anywhere there's an Internet connection and has a check-out model that avoids conflicts between versions. CVS can be used for programming source files, literary source files, and our own documentation.

4.6 User Interface Prototype

This section describes the first version of the user interface prototype that was built. Such a prototype is built to form as a basis or a sample during coding. During the design of the prototype interface, we verified that all interface requirements from section 3.4.2 are adhered to. As such, during further implementation, the programmers can concentrate on technical approaches used to make the interface viable, rather than merely concentrating on incorporating and dealing with the user and functional requirements. Obviously, after completion of coding, a reflection (validation and verification) must still take place.

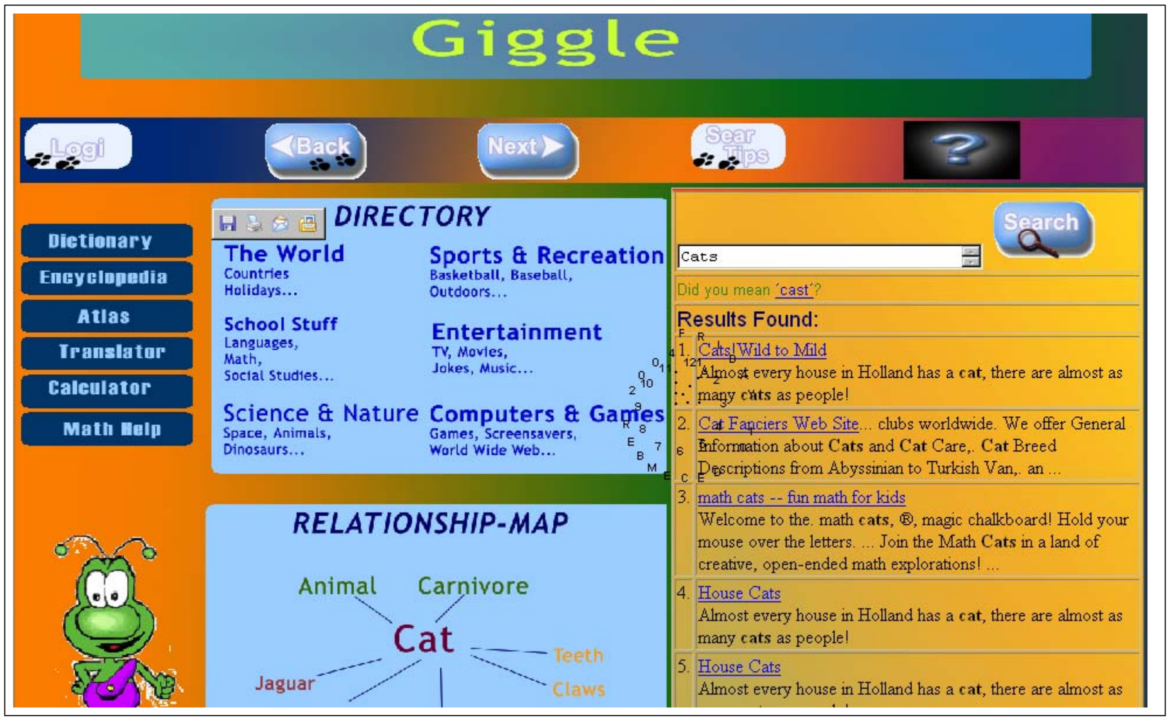
The prototype interface is composed of several screens.

- Main screen: The first screen the user sees when entering the site. This screen links to all other screens and incorporates all user search functionalities. See figure 4.14.

- Main screen for logged in user: Same as the main screen except for the fact that it features additional functionalities such as user-defined links and save/store search. See figure 4.16.

- Login screen: Capability for a user to login with user name and password. A new user is granted the possibility to create a new account. Also features possibility for user to authenticate himself in the case he has forgotten his password. See figure 4.17.

- Preferences screen: This screen allows the user to change his or her preferences. See figure 4.18.



Did you mean 'cast'?

Results Found:

1. Cats! Wild to Mild
Almost every house in Holland has a cat, there are almost as many cats as people!
2. Cat Fanciers Web Site... clubs worldwide. We offer General Information about Cats and Cat Care,. Cat Breed Descriptions from Abyssinian to Turkish Van,. an ...
3. math cats -- fun math for kids
Welcome to the. math cats, ®, magic chalkboard! Hold your mouse over the letters. ... Join the Math Cats in a land of creative, open-ended math explorations! ...
4. House Cats
Almost every house in Holland has a cat, there are almost as many cats as people!
5. House Cats
Almost every house in Holland has a cat, there are almost as

Figure 4.14: Preliminary Interface - Main Page

- Help screen: This screen features a user manual.
- Result document screen: The retrieved document is shown in a separate screen with search terms highlighted and options for the user to indicate if it is a relevant or irrelevant document. See figure 4.15.

4.7 Early Usability Evaluation

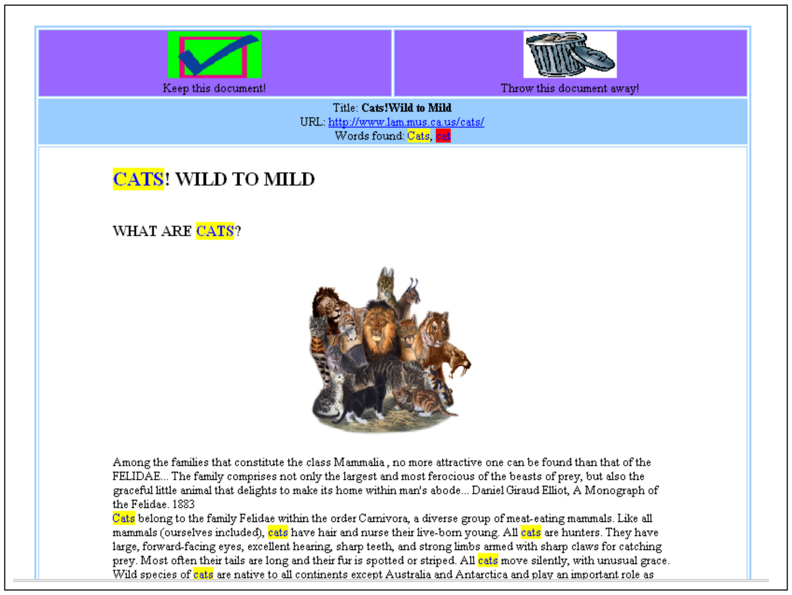
4.7.1 Functional Review

In section 4.3.1 we establishes a list of user-enabling functionalities which were to be embedded in the design in order to be sure that the product being built will be successful.

User-enabling Functionalities

As a prototype has been built, which forms the basis for the final product, we can now easily check if all user-enabling functionalities have been implemented in the prototype. Table 4.2 describes the results.

As you can see, all requirements have been incorporated with the exception of the advanced search. Further investigation has led us to believe that children have just extreme difficulty with understanding boolean searches that we wish not to implement this feature. This was previously indicated during the Requirements Analysis phase, however with new information that has come to light while spending time with (and teaching) children we have come to



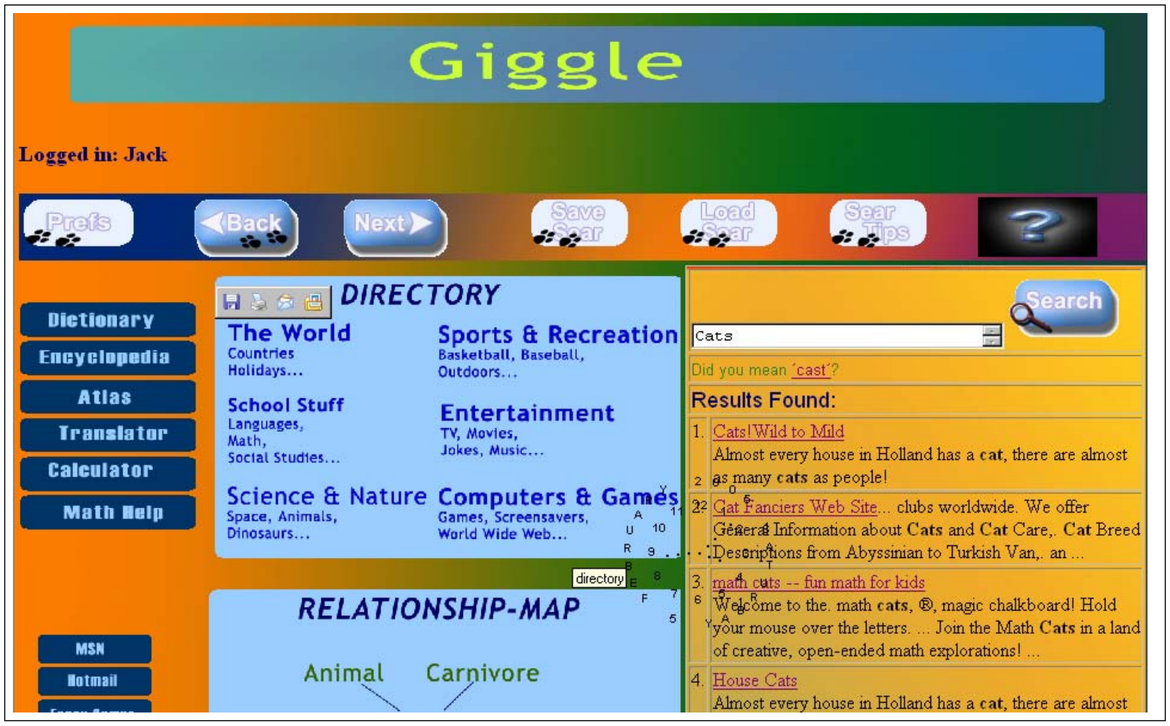
Title: **Cats!Wild to Mild**
URL: http://www.lam.mus.ca.us/cats/
Words found: Cats,

CATS! WILD TO MILD

WHAT ARE CATS?

Among the families that constitute the class Mammalia , no more attractive one can be found than that of the FELIDAE... The family comprises not only the largest and most ferocious of the beasts of prey, but also the graceful little animal that delights to make its home within man's abode... Daniel Giraud Elliot, A Monograph of the Felidae. 1883
Cats belong to the family Felidae within the order Carnivora, a diverse group of meat-eating mammals. Like all mammals (ourselves included), cats have hair and nurse their live-born young. All cats are hunters. They have large, forward-facing eyes, excellent hearing, sharp teeth, and strong limbs armed with sharp claws for catching prey. Most often their tails are long and their fur is spotted or striped. All cats move silently, with unusual grace. Wild species of cats are native to all continents except Australia and Antarctica and play an important role as

Figure 4.15: Preliminary Interface - Result Document Page



Cats

Did you mean 'cast'?

Results Found:

1. Cats!Wild to Mild
Almost every house in Holland has a **cat**, there are almost as many **cats** as people!

2. Cat Fanciers Web Site... clubs worldwide. We offer General Information about **Cats** and **Cat** Care,. **Cat** Breed Descriptions from Abyssinian to Turkish Van,. an ...

3. math cats -- fun math for kids
Welcome to the. math **cats**, ®, magic chalkboard! Hold your mouse over the letters. ... Join the Math **Cats** in a land of creative, open-ended math explorations! ...

4. House Cats
Almost every house in Holland has a **cat**, there are almost

Figure 4.16: Preliminary Interface - Main Page After Login

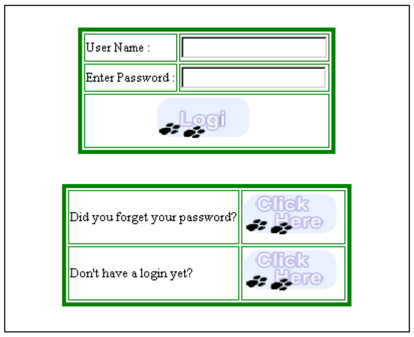
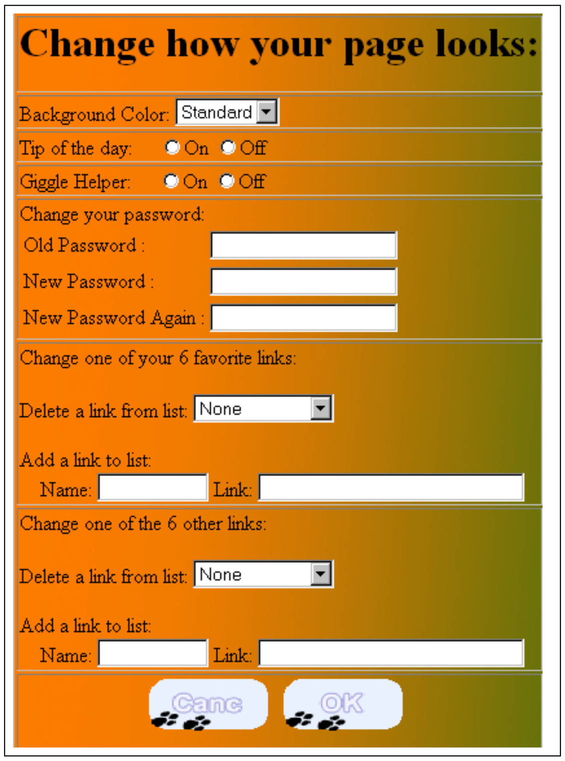


Figure 4.17: Preliminary Interface - Login Page


Change how your page looks:

Background Color: Standard

Tip of the day: On Off

Giggle Helper: On Off

Change your password:

Old Password :

New Password :

New Password Again :

Change one of your 6 favorite links:

Delete a link from list: None

Add a link to list:

Name: Link:

Change one of the 6 other links:

Delete a link from list: None

Add a link to list:

Name: Link:

Figure 4.18: Preliminary Interface - Change Preferences

Description	**Requirement**	**Result**
Instructions/Help	F1, F3, F9, NF1	✓
Metaphor	F2, NF1, NF2	✓
Query input	F4, F5, F6, F9, NF1	✓
Directory	F3, F4, F5, F9, NF1	✓
Relationship-map	F4, F5, F9	✓
Results list	F7	✓
Result document	F7	✓
Reversal of Actions	F8, NF1	✓
Personalization	NF2, NF4	✓
Links list	NF2	✓
Time and date	NF2	✓
Advanced search	F9	X

Table 4.2: Test results for user-enabling functionalities in prototype.

learn that the problem is larger than we had previously expected. Designing the functionality will require much more effort than merely offering several input boxes (possibly a form of education). Because this falls beyond the scope of this project, we have chosen to drop this secondary requirement. Because requirements analysis has shown its necessity, we do not drop the requirement for good, but postpone it to future elaboration of the product and have included it in the recommendation list.

Supporting Functionalities

We must also verify if all the supporting functionalities have not been verified above. Three types of requirements must be dealt with here, functional, non-functional and business requirements.

- *Functional Requirements:* As functional requirements yield a particular system-behavior they can be described using a model. We verify to ensure that each requirement is modelled. To ensure that it has been modelled correctly we consult the system walkthrough description (see appendix B.2), check if models are consistent and not conflicting, and complete verification by performing mental verification.

- *Non-functional Requirements:* Non-functional are not applicable to modelling as they do not exhibit a particular behavior. However, they have been incorporated in system-relevant architectural decisions, ensuring correct architecture decisions are made based on them.

- *Business Requirements:* Business requirements are not applicable to modelling for the same reason as non-functional requirements, and are dealt with in the same (architectural-based) manner. However, one requirement has not been dealt with. Building awareness is not a requirement that must be realized by the tool, but a process requirement which must be realized by the project owners by completing a separate process. In cooperation with the client we must establish a plan in order to achieve this, and test if it has been achieved during the Acceptance and Finalization phase (see section 8.4.1).

Description	**Requirement**	**Models**	**Result**
Query modification	F5, NF3	Section 4.4.5 Section 4.5.2 Figure 4.3 Figure 4.10	√
User safety	NF2, NF3, NF4	Section 4.4.5 Section 4.5.2 Figure 4.3 Figure 4.9	√
Document Format Types	NF5	Section 4.4.5 Figure 4.3	√
Hardware and Software	NF5	Section 4.5.3	√
Performance	NF6	Section 4.5.3	√
Awareness	B1	-	X
Multi-linguistic	B2	Section 4.4.5 Figure 4.3	√

Table 4.3: Test results for supporting functionalities in design.

In order to verify all these functionalities, table 4.3 functions as a checklist. It describes all the requirements, in which models they have been incorporated or, if modelling is inapplicable, in which section they are dealt with. We furthermore indicate whether we find the modelling (or description) significant enough to proceed to implementation. That is to say, the required functionality has been broken down described in such a way that a programmer should encounter no issues while coding it. This last aspect is important because the models have a different level of specificity which may not all be low-level enough to properly indicate implementation, we thus must critically view and evaluate the descriptive power of the models.

As we see, all but one of the requirements (raising awareness, described above) have either been modelled, or if not applicable to a model (such as non-functional requirements) have been described and incorporated in system-relevant architectural decisions, with exception of one requirement. In all cases, we have ensured that the functionalities will be taken into the implementation phase, either by directly modelling them or by choosing an architecture which makes them possible. After implementation, the following requirements engineering phase, we test to ensure that all these requirements have indeed been adhered to during coding and implemented in the product.

4.7.2 Cognitive Walkthrough

Solution Overview

The Giggle application is an internet search engine that supports taking an input from the user (as a keyword query input or selection from a directory or relationship-map) and returning a set of the most relevant documents out of the database. It furthermore is equipped with many features such as the capability to save/store temporary search states and maintain a favorites list.

Method

The cognitive walkthrough evaluation involves observing user representatives performing selected tasks using a prototype of the available interface windows for an application. A follow-on design session with user and development representatives occurred to resolve some of the issues that arose during the evaluation. Most often interfaces drawn on paper are used for the cognitive walkthrough. However, because our user and participants are children and may not have the ability to see beyond that, we have implemented an interactive interface prototype for this purpose.

Participants

Five children representing the users participated in the evaluation. These participants had an average age of 11 years and had each previously been performing the tasks tested in this evaluation for at least a few years. Furthermore, we asked two parents to participate in the evaluation. Though they are not the users for which the product is made, they are likely to be involved in the use, possibly because a child wants assistance in a search.

Tasks

Four scenarios were performed:

1. Executing a search by query;
2. Executing a search in directory;
3. Executing a search using the relationship-map;
4. Logging in and saving a search state.

Usability Impact Ratings

Usability issues or questions are rated according to the level of impact they have on successful task completion and user satisfaction. The following levels of impact are defined:

- *High:* Preventing the user from successfully completing the task, will cause extreme user dissatisfaction or make it easy for the user to make errors.

  Action: Must be corrected or resolved prior to deployment to enable task completion and avoid extreme user dissatisfaction

- *Moderate:* Causes the user difficulty but task can be completed, will cause user dissatisfaction.

  Action: Should be corrected or resolved prior to deployment to facilitate task completion and avoid user dissatisfaction.

- *Low:* Minor problems that do not significantly affect task completion, however may cause some user dissatisfaction, especially when considered in aggregate.

  Action: Should be corrected or resolved prior to deployment to avoid user dissatisfaction.

Procedure

The participants took part in the evaluation individually. The participant was introduced to the concept of usability evaluation and its goals. The participant was distinctly told that we were testing the software design, not the participant. The participant was informed that he or she would take part in four scenarios to evaluate a software design. The participant was encouraged to think aloud as he or she walked through the scenarios and given time to ask questions. The participant was also reminded that the software was in early design, so significant changes could be expected by the time the final software product would be implemented.

Results

The results of the walkthrough findings are described in table B.1.

Next Steps

The user interface design for Giggle has made an excellent start. It is recommended that the following additional usability activities be performed:

- Conduct additional design sessions to layout and review the windows for the various functions;
- Develop a more complete design prototype of the directory and the relationship-map functions which are clickable so that these functions can also be tested with the appropriate users.
- Conduct additional usability tests with the new prototypes.

4.7.3 Heuristic Review

Because of its similarity to cognitive walkthroughs, only the method and results sections of a heuristic review are documented here.

Procedure

A heuristic evaluation was undertaken based on the primary objectives of the product and its intended users. The types of heuristics used have been lifted from section 3.4.2 (mostly from requirement NF2)from the Requirements Analysis which was based on an extensive literature study.

In short, the heuristic types are:

1. *Visibility of system status:* The system should always keep users informed about what is going on through appropriate feedback within a reasonable time.
2. *Match between system and the real world:* The system should speak the user's language, with words, phrases, and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

3. *User control and freedom:* Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

4. *Consistency and standards:* Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

5. *Error prevention:* Even better than a good error message is a careful design that prevents a problem from occurring in the first place.

6. *Recognition rather than recall:* Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

7. *Flexibility and efficiency of use:* Accelerators, unseen by the novice user, may often speed up the interaction for the expert user to such an extent that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

8. *Help users recognize, diagnose, and recover from errors:* An error message should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

9. *Help and documentation:* Even though it is better if a system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

A checklist capturing these heuristics was used to aid evaluation (see the following section). Note that performance, although a secondary contributor to usability, was not considered in this evaluation.

Results

Each usability problem uncovered was noted and categorized according to the heuristic types described above. A recommendation for design change was specified. The results are summarized in tables B.2 and B.3.

4.8 Conclusion

The goal of this chapter was to formally translate all requirements into a complete design which can in turn easily be translated to code, in order to successfully design and implement an easy to use tool that supports children in their search for information on the Internet.

The list of requirements and the original use case scenario from section 3 formed the basis of this phase. The use case model was modified to reflect all the new requirements posed. These were divided into two groups depending on whether they were functions that must be offered to the user (and thus modelled in the use case diagram), or underlying requirements for the system (and thus most be dealt with in an alternative manner). One by one the requirements were translated into a design. Afterwards, in section 4.7.1 verification took place to ensure all the requirements had be incorporated. Furthermore, usability evaluation took place to ensure the design was feasible for what the users intend it to do, and thus capable to perform the functionalities expected from it as stated in the project goal. Also, the user interface prototype was verified in cooperation with the intended end users to ensure that all functionalities were available, clear and the interface could be used for the purposes for which it was intended, the primary goal of this project.

To ensure that the design can easily and correctly be translated to code, a well known method called Rational has been used. Once a design is described in this proven formal design model, it can iteratively be described in more detail until such a low-level description exists that it can easily be translated to code on a one-to-one basis. However, creating such designs is very time-consuming and we did not complete the process to its smallest details. We used the model to describe overall functionality and separate these into small well-defined subcomponents. This greatly reduces complexity, as each unit can be implemented and tested separately. After coding, the design can be used to ensure no design elements have been omitted in the code.

As the design passed the heuristic evaluation and cognitive walkthrough tests, and furthermore it has been tested that all primary and secondary requirements have been incorporated in the design, we conclude that our design is a correct translation from the requirements posed in section 3.

During the design analysis phase, we have dropped a secondary requirement (advanced search) in the light of new information. We have learned, as we are still working with children, that the functionality posed by the requirement is far too complex for children to work with if designed and implemented straightforwardly. The functionality could only be useful to a child if it is supported by education. As education falls beyond of this project, and implemented a feature children cannot work with is not included in the project goal, we choose not to design nor implement this feature at this point in time. However, as it was a secondary requirement, it has been placed in the list for future recommendations at which time more research can be done into the topic.

While testing the results of the design phase, i.e. whether all requirements had been mapped to the design, we realized that a requirement could and had not be modelled nor designed. The requirement to adhere awareness, must be fulfilled during an individual process. Whether this has been achieved must be tested in an alternative manner during the Acceptance and Finalization phase.

We have seen however that during the design phase we have not only have had to define the requirements more exactly into a sound specification, but also have added our own requirements to make implementation possible. An example is breaking down the code into smaller units such as classes, making implementation and testing more manageable.

4.8.1 Evaluation

In order to verify that the design models we had built were adequate, we took various steps. First of all, we separated the requirements into groups, depending on what type of a model could be used to represent them (as there a plethora of models which can be used to fully represent all functionalities of a software system). For each list, we incorporated the requirements into a design, and then afterwards, verified that all requirements had indeed been incorporated into the design. We showed our models to programmers at IBM and our supervisors at the university to get feedback and determine if we had left anything out.

We built a prototype interface which our users could use and click on in order to determine if it contained all the features expected, and furthermore if the functionalities were self-explanatory, easy to work with, and the interface adhered to their preferences. This prototype testing was only carried out on a handful of children which does not give an adequate reflection of our entire user group. However, the children used have been carefully selected to represent the user group adequately (characteristics such as diverse backgrounds, age, sex, and computer experience). We showed our interface to two parents for feedback. We also showed our prototype interface to our client to ensure that this was what he expected from the software. Feedback from the users, parents and our client has been incorporated into a new interface design which has also been tested with users. After having performed various feedback loops, we considered our interface prototype as validated and could use that as a basis for further design.

We met with two experienced IBM software programmers to discuss our models. They not only gave us feedback on the models, but were also able to help us make architectural decisions based on the specification of the tool we wished to build, the future expectations of the tool, and available architecture at IBM. Feedback was incorporated into the models, and architectural decisions were based on the expertise and advice of the software programmers.

Throughout this stage we received very much feedback from various sources. As many sources have been consulted, feedback was incorporated into the model at each stage, and all people consulted agreed on the model, we feel we have devised a complete and valid design which describes all requirements posed.

Chapter 5

IR Research

5.1 Executive Summary

Led by existing information retrieval research, this section discusses algorithms and methods which could increase the retrieval performance of Giggle. These methods were evaluated and tested. Section 5.5 gives an overview of the results. Not all (positive) research results have been implemented in Giggle because they must first be performance tested more adequately, however they do form a good starting point for future elaboration.

5.2 Introduction

The Project Plan (section 2) established a computer science goal underlying the academic research on which this thesis is based. This academic research is imperative not only to show that the project owners are capable of attacking and dealing with new problems, yet also distilling information from other people's scientific research, and able to consider different approaches and methods and applying them to the problem at hand in search of a solution.

Computer science aspects pertaining to the graphical user interface have previously been investigated and dealt with by establishing a list of guidelines, and therefor will not be given any attention in the remainder of this chapter. This chapter describes the research we completed in an attempt to improving the functioning of Giggle using aspects from the computer science field of Information Retrieval.

5.2.1 Goal of IR Research

The goal of information retrieval research is to identify methods or procedures that will aid in the goal to successfully design and implement an easy to use tool that supports children in their search for information on the Internet.

Because the goal stated above is very broad, we have decomposed it into two subgoals:

1. Determining characteristics and methods to design and implement an application which is intuitive enough for children to work with yet strong enough to aid them in their search for information on the Internet

2. Determine how, if it is possible to adhere to the user's needs by increasing retrieval precision, delivering relevant documents and filtering irrelevant documents.

As this document describes and proposes algorithms from the field of information retrieval, it has been written only for people interested in the field and our research. As such, it can be read by our supervisors to justify that we are capable of dealing with new problems and solution with an academic approach. Furthermore, it can be used by any other computer scientist who wishes to implement algorithms for a similar purpose.

5.2.2 Research Process

In the previous section we distinguish between two subgoals. The first pertains to user interfaces including navigation models. Because the interface is an aspect which must be incorporated in the design of the model, research pertaining to this aspect has been completed prior to the design phase during which we investigated relevant research. The results of that investigation have been described in section 3.4.2, and have been incorporated in the preliminary interface prototype (see section 4.6) which was thereafter evaluated for usability (see section 4.7). The results were positive, and the investigation to this subgoal has therefor been concluded.

To answer the second subgoal has itself been divided into several steps:

- Extensive literary research into existing relevant relevant research;
- Evaluating the advantages and disadvantages of different methods and algorithms, determining which would be most appropriate in helping us achieve our goal;
- Implementation of the algorithms, and testing using the prototype to determine if it could truly be of use to us;
- Creative experimentation. Some open issues remained that we wished to deal with, for which we could not find adequate literature research describing possible solutions. For these issues the real fun work started. We devised our own plans and methodologies and tested if they could possibly help. Not surprisingly, most of these failed. But failed or not, we actually completed research and were able to pull our conclusions from our attempts.

The first section (Section 5.3) describes some ground principles of relevant aspects of important research in the field and defines some of our assumptions to avoid miscommunication by unambiguous definitions. The second section (Section 5.4) describes which research and investigations we performed and tested on the Giggle prototype and discusses its results and usefulness in the Information Retrieval field. In the last section (section 5.6) we map the general results back to our goal.

5.3 Literary Research

This section gives a global introduction to the field of Information Retrieval by summarizing the literary research that we completed in line with incorporating and improving information retrieval aspects for Giggle.

5.3.1 Retrieval Performance

In our goal we speek of improving retrieval performance, we thus need methods for determining and improving relevance. First, let's define the performance of retrieval relevance.

Relevance is the measure of how well a retrieval results answers a particular user query, and thus their information need. The main purpose of a search engine is to maximize relevance while minimizing burden to the user. In considering retrieval performance evaluation, and thus the relevance result we distinguish between the following two measures:

- **Recall:** The fraction of all the relevant documents that have been retrieved:

$$Recall = \frac{|Relevant\_docs\_retrieved|}{|Relevant\_docs\_in\_entire\_collection|} \tag{5.1}$$

- **Precision:** The fraction of the retrieved documents which is relevant:

$$Recall = \frac{|Relevant\_docs\_retrieved|}{|Retrieved\_docs|} \tag{5.2}$$

Much research has gone into improvement of relevance [106]. What stands central are efforts to increase retrieval precision and recall simultaneously, however, practice has shown that these do not increase simultaneously. Precision and recall can be used to evaluate quantitatively both the quality of the overall set of retrieved documents as well as the breadth of the retrieval algorithm.

5.3.2 Retrieval Relevance

Now that we've defined performance, we must determine how performance issues relate to the relevance of documents to our user group.

We distinguish between different sets of documents being benign (as opposed to harmful, as described in chapter 3) and being relevant to the user according to their information need. During the requirements analysis phase, the stakeholders have indicated that retrieving documents that retrieving primarily benign documents is preferred, and if that means that not all documents pertaining to the information need are retrieved, so be it! As such, understanding that precision and recall cannot be boosted simultaneously, our preference goes for a high precision and the highest recall possible under that preference. The different sets of documents have been described in section 3.4.2 (see also Figure 3.2).

5.3.3 Retrieval Improvement

There are a variety of approaches for improving initial query formulation, as an attempt to increase the precision of the retrieved results, through methods of query expansion and term reweighting. The following six groups can be distinguished and will be explained in more detail in the following sections:

1. User relevance feedback
2. Local analysis
3. User profiling

4. User-group profiling
5. User input

5.4 Information Retrieval Methods

This section describes different topics for which we sought answers or solutions, how we approached them, and what the conclusion of our investigation was.

5.4.1 Query-Document Relevance

In order to simplify comparison between documents and queries, both are translated and stored as vectors. Similarity of document and query vectors indicate relevance (ranking). As such, we define a document of perfect relevance to a query when both vectors are identical; obviously, this is often not the case. Distance between vectors is measured using the $\cos\theta$. Identical vectors have a $\cos\theta = 1$, orthogonal vectors have a cosine of $\cos\theta = 0$, so all relationships can be mapped into this range.

5.4.2 Reweighting for Collection

There is not much public information about the specific algorithms used by current search engines for reweighting techniques, we must thus rely on our own expertise and opinion in this case. A rather simple, yet very effective vector reweighting technique is the TF-IDF (Term frequency - inverse document frequency) schema. The conversion of a term-frequency weighted vector into a vector with a specified weighting scheme occurs by altering the document weight based on information about term frequencies in the entire collection (done individually on each term). Formally, let N be the total number of documents in the collection and n_i be the number of documents in which the index term appears, then:

$$w_{i,j} = tf \times idf = f_{i,j} \times \log\left(\frac{N}{n_i}\right) \tag{5.3}$$

If the new term weight is equal to 1, only inverse-document-frequency is considered. As a result of this weighting scheme, terms occurring in all documents of the collection have no distinctive power and are not used to primarily describe a document (content). This schema can be used to strip ’meaningless’ (i.e. terms without distinctive power) terms from a document.

5.4.3 User Relevance Feedback

Based on feedback from the user, relevance feedback is the most popular query reformulation strategy. The user is presented with a list of retrieved documents. He or she marks which documents are relevant, upon which the system selects important terms, expressions and phrases from the marked documents. These can be used to create a new query and rerun the query resulting in a document list with a higher relevance than the initial results list. Though this technique requires user-assistance, an automatic variant also exists whereby merely the top ten retrieved documents are taken to be relevant.

Query expansion can be used for this approach. Users having difficulties thinking of terms to formulate queries can benefit greatly from query expansion methods in which relevant documents are analyzed and new terms are added to the original query. The modification of term weights based on the user relevance judgements is called term reweighing. Because relevant documents resemble each other, so do their term-weight vectors. The idea is to reformulate the query vector so that it moves closer to the term-weight vector space of relevant documents and away from that of non-relevant documents.

Query expansion methods have been studied for a long time in the Information Retrieval field. Recent researches have shown that query expansion is a useful to improve retrieval performance with general collections, yet seldomly explored as a technique for commercial use, especially Web search engines ([106]). The Rochio formulation is a classic technique based on the vector model which combines query expansion with term reweighing. It is rather simple because the modified term weights are computed directly from the set of retrieved documents, yet yields good results (observed experimentally) because the modified query vector reflects a portion of the intended query semantics, however, no optimality criterion is adopted.

Many experiments have shown that user relevance feedback is an extremely useful and powerful method of improving the retrieved document set [106]. Traditionally, relevance feedback provides the user with a button or link to ”find more documents like this one”, usually located next to each document on the results page. The relevance feedback mechanism typically issues a new query that is created from the most frequent terms in the document. This feature, present in the early days of Internet search engines, has disappeared from commercial usage. The problem is that from a results page, it is not yet known whether a document is relevant or even what it is about. For this reason, using relevance feedback from a results page is unpredictable, and negative user perceptions of this feature have led to its elimination. However, the idea of user relevance feedback can lead to positive results. We therefor propose to find another manner in which the user can indicate relevance, already having seen the document at hand.

- **Hypothesis:** If the user indicates certain document relevance or irrelevance, reweighting the query using information from this document results in a query which more resembles the user's information need and thus increases retrieval precision.
- **Goal:** Tune query based on user relevance feedback in an attempt to increase retrieval precision.
- **Approach:** Due to its simplicity yet remarkable retrieval results, we have chosen for the implementation of the standard Rochio algorithm.
  - Transform the original query into a tf-vector
  - Given a relevant document, transform this into a tf-vector
  - To calculate the modified vector, add the document vector to the query vector.
  - Repeat the last two steps for each relevant document that has been indicated, each time using the modified query as the input.
  - Given an irrelevant document, transform this into a tf-vector. If the resulting tf value is smaller than 0, remove the tf value from the vector (we don't want negative values in a tf vector, and for efficiency purposes do not store terms of 0 values in the vector).
  - To calculate the modified vector, subtract the document vector from the query vector.

  - Repeat the last two steps for each irrelevant document that has been indicated, each time using the modified query as the input.
  - Normalize the modified query vector
  - Use this query vector for determining retrieval relevance scores of new documents

- **Result:**

  The method was implemented. Assuming that the user adequately indicates document (ir)relevance, testing has shown that the modified query does indeed resemble the information need more adequately. However, no testing has yet been done to determine if retrieval precision does indeed increase.

  Parameter tuning of the Rochio algorithm has not taken place. Literature has indicated that tuning must be adapted according to the set of documents in the collection. As the collection was not complete at the time of implementation, standard values of 1 have been used.

  Due to the addition of query terms, query becomes much larger than the original query. We expect this to hurt system performance. Adequate testing should be completed to indicate if this is true or not. However, the addition of query terms has shown to be advantageous because of the addition of relevant (synonymous) terms which the user would otherwise not think of.

- **Discussion:** Preliminary tests have shown a modified query that more resembles the information need of the user.

  Relevance feedback is especially useful for our user group because it:

  - shields the user from the details of query reformulation process because all the user has to do is indicate which documents are relevant
  - breaks down the searching task into a sequence of small steps which are not only easier to grasp but also lower the cognitive burden on the user
  - provides a controlled process designed to emphasize relevant terms and de-emphasize non-relevant terms.

  However, users are reluctant to use relevance feedback. Our design proposes to implement user relevance feedback in such a manner that the users are forced to make use of it. After the primary retrieval run, the users are shown a list of between five and seven retrieved documents (GUI and cognitive research on children has shown that this is optimal, however the precise number must yet be determined). The user can indicate which documents are interesting and by doing so save them in a separate 'relevant documents' list (which can always be retrieved during a later session and furthermore gives an overview of the retrieved relevant documents). Furthermore, the user can indicate an irrelevant document by deleting this from the list. Upon doing so, the relevance information indicated by the user up to that moment (which documents had to be saved (were relevant) and which were to be deleted (irrelevant)) will be used to retrieve a new document and thereby to refill the empty location in the list. If the user wishes to view more documents than merely those five or seven shown, the user is forced to indicate which of the previously retrieved documents are relevant and which are irrelevant. It must be clear to the user that marking a document as relevant has an affect on his or her results. Incorrect use of this functionality will dramatically deteriorate the quality of retrieval results.

However, query expansion methods also have their pitfalls. Experiments have shown that it can greatly improve retrieval results if the correct expansion terms are chosen, however, it dramatically hurts the results if incorrect terms are chosen. This method will only work if the user adequately indicated the relevance of a document. Incorrect indication will have disastrous effects. Furthermore, the product must be thoroughly tested to ensure that the query expansion method that is to be implemented actually improves the retrieval results. Adequate gui design choices must be implemented to aid the user in this proces.

5.4.4 Query Concepts

In order to improve the user input, methods to improve the translation of the user's information need into a query which the system can use for retrieval searches may be useful. Obviously, we must attempt to burden the user as little as possible, which can be achieved by incorporating mechanisms with which the system can translate a user input into an appropriate query.

Local analysis uses clustering techniques to automatically (without user assistance) expand the query based on information derived from the set of documents initially retrieved (local set of documents). The idea is to determine a larger cluster of relevant documents by adding thesaurus-like relationships such as query synonyms, stemming variations, and collocations.

Semantic relations, so called searchonyms, can be used to help choose the right descriptors and expand the query:

- synonyms
- stemming variations
- related terms, those close to (empirical distance) query term in the text
- spelling variations
- hypernyms, hyponyms, and meronyms

Research has indicated that this technique is not cost effective for web applications, but rather for smaller collections, because it takes too much time to calculate and furthermore requires a lot of CPU time. However, because the collection of documents which can be retrieved (according to user requirements) is only a subset of the web and most probably must be determined a priori, this technique may be applicable to our system.

The conceptual model is a technique which analyzes documents and determines important concepts, such as nouns or noun groups. Using relevance feedback we can determine which documents are relevant, and by building a conceptual model on the entire collection or corpus of relevant documents, it allows us to determine which concepts are relevant. The concepts found can be arranged in a concept lattice. For reasons of complexity, several concept lattices are more appropriate than merely one. Furthermore, concepts can be generalized or broadened by mapping them to concept groups. The concepts found can be used to expand the query.

- **Hypothesis:** Applying the same idea of ”wide-concepts” to queries as has been done with documents will increase retrieval performance.

- **Goal:** By means of local analysis, determine which concepts (and their one-level higher hypernyms) appear in a query and use that information to retrieve documents to that query.
- **Approach:**
  - Given a query create a term-frequency vector using not only every term, but also every consecutive term pair;
  - For each value in the term-frequency vector (thus document terms and document term pairs) determine the hypernym of that term and add it to the tf-vector with a weight of half the original term;
  - Use this new query for document retrieval.
- **Result:**

  The preliminary results showed to be poor. Though the approach increases recall, it dramatically hurts precision. The query is expanded to such an extent that it loses its precise meaning. A query input "cat" can yield a result about dogs only because they are both mammals, and the document may use the term mammal rather frequently.
- **Discussion:** Results showed to be very poor. Research aborted.

5.4.5 Document Concepts

Similar to the local analysis for query concepts, we were curious if the same could be done for the concepts in documents.

- **Hypothesis:** Under the assumption that if a term frequently occurs in a document, then that document is about that term, we consider the same for so called "wide-concepts", a concept or relevant concept occurring frequently in a document indicates that the document is about that concept.
- **Goal:** By means of local analysis, determine which concepts (and their one-level higher hypernyms) appear in a document and use that information to retrieve documents given a query.
- **Approach:**
  - Given a document create a term-frequency vector using not only every term, but also every consecutive term pair;
  - For each value in the term-frequency vector (thus document terms and document term pairs) determine the hypernym of that term and add it to the tf-vector with a weight of half the original term;
  - Use this new document tf-vector for document retrieval.
- **Result:**

  The preliminary results were very positive. Authors often incorporate synonyms and terms slightly higher in hierarchy to avoid term-repetition and make a document more interesting to read (e.g. a document primarily about cats will incorporate the concepts cat, house-cat and jaguar). The combination of these terms occurs more frequently than that of other terms; have a look at the following example:

”Felines are interesting creatures. The feline family has many members, but what they all have in common is that they like the same prey. Jaguars like mice. House cats also like mice.”

The traditional tf-vector would describe this document to be equally about felines, cats and mice. However, this version will see that felines and cats are related, and that the document is more likely to be about cats than about mice.

Testing has shown that this method works rather adequately. We have incorporated this method in Giggle to test if it really works in practice. Further testing is required to ensure that the results are true, especially on children's documents

- **Discussion:** Characteristically of children's documents is the use of short sentences with many synonymous and hypernomous terms. Using the method of expanding a document with it's ”wide-concepts” to increase retrieval recall, yet also slightly increase retrieval precision has shown to be advantageous.

5.4.6 Directory Description

We assume that documents which can be attained through popular children's sites which have been manually-censored by experts, are safe and white listed. Furthermore, these documents have manually been placed in directory listing with care. Using the documents in each of the directory lists, we can derive information with which we can automatically categorize unseen documents.

- **Goal:** Automatic categorization of a new document.
- **Hypothesis:** Given a set of vectors which gives a general description of documents in each category, it can automatically be determined to which category or categories a document belongs to.
- **Approach:**
  - Using existing childrens directories and results from the requirements analysis, determine the highest-level categories which are to make up the directory;
  - Gather a set of childrens documents representative for each chosen category;
  - For each document in a category, create a vector which describes the sets of words (and their frequencies) of each document;
  - Normalize the vectors;
  - Calculate tf-idf on each of the vectors for the purpose of noise removal and terms which have no distinguishing powers.
  - Find the average vector component values for documents belonging to a particular group, resulting in a vector which (on average) describes the entire category.
  - Compare the resulting vectors of each category (by means of similarity) to check for adequate differences and thus distinguishing features of each vector.

  - Given an unseen document, compare its normalized vector to that of each category, in the same manner that a query vector is compared to a document vector. The result is a relevance score. If the relevance score is higher than a particular threshold, the category name is temporarily stored. After having compared the vector to each category, the category with the highest relevance score is the category to which the document most likely belongs to. In the case that the top two (or more) categories have a very similar score, the document belongs to both (or more) categories (a desired result as documents may belong to more than one category).

- **Result:**

  This method has been implemented and tested on a global level. Though more systematic and extensive testing is required to be completely sure that it works as expected, the preliminary tests show that the results are promising. The fact that a list of relevance score is yielded has shown to be interesting and useful. Often documents belong to more than one category if graded by the important concepts in the document, and thus a method of categorization must allow so. Furthermore, the fact that only relevance scores above a certain threshold are permitted has also shown to be useful. Documents not clearly pertaining to any category will merely not be categorized in the directory structure rather than simply being placed in the category of highest score. This is desired to ensure that no documents are classified that seem irrelevant to a category (have a very low relevance score).

- **Conclusion:**

  Because testing has shown that this procedure suffices, it has been incorporated in Giggle. The prototype can be used to further test the idea, yet the first results seem promising.

5.4.7 Stemming Document and Query Contents

- **Hypothesis:** Due to the nature of the English language, stemming each of the terms in a document will increase retrieval recall.

- **Goal:** To increase recall by mapping word tenses to the same grammatical form to allow for retrieval matching.

- **Approach:**

  - Translate the document to a tf-vector.
  - For each term in the tf-vector, replace the term by its grammatical stem.
  - Apply the same method to the query.
  - Use the stemmed document tf-vector and the stemmed query tf-vector to calculate retrieval results.

- **Result:** Due to the nature of the English language, stemming terms maps terms with similar meanings to one grammatical root (e.g. the terms ”interpret”, ”interpretation”, and ”interpreting” all have similar semantics and will all be rooted to the same variant).

Stemming terms to their grammatical roots not only shortens the tf-vector dramatically, yet broadens the mapping performance of the retrieval procedure because a query term can yield a higher range of documents. As such this can be useful when one inputs a query term in its plural form yet would also benefit from results yielded by its singular form. Our research has indicated that this is the case with our users due to the lack of linguistic knowledge that children have.

- **Discussion:** According to research this method can compress index size by at least 40% [106]. Though we did not evaluate the exact quantity, we have seen that this procedure does compress the index size dramatically. We must keep in mind though, that due to a loss of information (you are throwing our information) we lose precision. For example, there is no distinction between "the president" and "president" while on a semantic level the first is much more specific than the second.

5.4.8 Self-learning Relevance Score

Relevance is the measure of how well the indexed page answers a particular user query. When there are a large number of matches for a query, the search engine must rank the results by relevance score, sorting the results listing so that the pages most likely to be useful will appear first.

A typical technique for relevance ranking is weighting. It is a heuristic technique designed to improve the relevance ranking algorithms. The most common relevance weighting is based on term frequency in the document (see section 5.4.2). Many search engines also decide that pages with matching terms in the title, keywords, description or headings are more likely to be relevant than other pages, so they add a number to the score to reflect that in the relevance ranking. Other weighting schemes recognize whether all query terms are in a page, the location in the web site, the date, and so on.

User-Group profiling makes use of information from the global population of users to indicate particular relevance and preference aspects. Storing and analyzing user queries and results can help determine what our users find relevant documents (dominant concepts, document types, and document formats), which topics they search, what the preferred search strategies are, and types of input queries are used. This information can be used for document ranking purposes or reweighting schemes.

In the document ranking, favorable documents should attain a higher ranking than gray documents. Black-listed documents should obtain a negative ranking. Documents frequently viewed by children should get a positive ranking.

- **Hypothesis:** Children generally find the same types of concepts relevant.
- **Goal:** Determine which concepts appear in documents that children find relevant and use this information during ranking to offer more relevant documents to all other users.
- **Approach:**
  - Concepts are given a global (over the entire user group) relevance score, a value between 0 and 1. 1 meaning relevant, 0 meaning irrelevant. New concepts are given a score of 0.5. Black listed concepts are given a score of 0.

- When a user indicates the relevance of a document, or during training of the database with known relevant documents, determine which concepts prominently appear in that document.
- For each appearance of the concept in a relevant document, increase the rating of that concept according to a particular ratio.
- For each appearance of the concept in a irrelevant document, decrease the rating of that concept according to a particular ratio.
- Upon retrieving documents from the Internet by the spider, the term-frequencies in the documents are multiplied by their concept rates.
- During document-query matching, the relevance rating between query and document is completed as usual.

• **Result:**

Initial results were positive. What is especially interesting is that this method retrieves documents containing child-popular concepts relevant to the query over documents not containing those concepts. The method does not just rate child-popular documents, but matches them to the query.

The new weighted term-frequency value is determined during retrieval of the document by the spider. Because the system continues to learn, the score may not reflect the current relevance score. However, old documents are refreshed by the spider when their date expires, and upon refreshing document information in the database, the term-frequency vector is recalculated and reweighted again.

• **Discussion:**

Though initial results were positive, the system must collect sufficient data from the users prior to completing a full test. Until then, no test results can be determined.

We have considered the option to maintain a user profile. That is to say, store all concepts found in documents marked as (ir)relevant by one particular user. This information gives an indication of the interests and general information need and preferences of the user and can be used during subsequent searches as an approach to increasing precision of retrieved documents [123]. However, though this may be an adequate method for retrieval improvement with a general internet searcher, a child of our focus group is in a process of learning and metamorphosing. We propose that a child in this developmental stage is changing too rapidly to adequately construct a user profile that meets these changing needs, which would in turn only hurt the retrieval precision. For this reason we choose not to implement user profiling.

Though we explicitly choose not to implement user profiling, we can use information from the global population of users to indicate particular relevance and preference aspects. The longer the system is in use, the more it can learn from its users, deriving information to improve subsequent retrieval results.

5.4.9 Terms of Childrens Documents

• **Hypothesis:** Terms specific to childrens documents are only a subset of terms specific to general documents.

- **Goal:** Database optimization by means of global analysis. To be able to restrict the lengths of wordlists used for determination of synonyms, spelling errors, hypernyms, etc.

- **Approach:**

  - Gather a set of documents representative for childrens domain;
  - For each document, create a vector which describes the sets of words (and their frequencies) of each document;
  - Normalize the vectors;
  - Adding all vectors together will yield a vector describing the document collection, and thus a list of all words which appear in childrens documents;
  - Compare the resulting list to the list of all WordNet words to determine if possibly some terms can be discarded from the database because they are merely not used.

- **Result:**

  The method has been implemented, yet the results were so large that it was difficult to draw any conclusions from it. Due to a lack of time, and the fact that this is merely a database optimization and not necessarily prerequisite to the functioning of Giggle, it has been aborted.

- **Conclusion:**

  Insufficient results. Research aborted.

5.4.10 Terms Categories of Childrens Documents

- **Hypothesis:** The categories to which terms appearing in childrens documents belong to are merely a subset of all categories existing in WordNet.

  Obviously, restricting the contents of wordlists in the database also leads to restrictions in the functioning capabilities of the final program. Terms or categories which are eliminated at this point in time can not be referenced to at a later time, and thus may have an impact on search precision and retrieval. However, because constraining the lists can result in a dramatic improvement in performance (for the same reasons of improvement that we use stop word removal), we suppose that such findings are very much of interest.

- **Goal:** Database optimization. To be able to restrict the number of words stored in the database by being able to remove entire categories which never appear in childrens documents.

- **Approach:**

  - Gather a set of documents representative for childrens domain;
  - For each document, create a vector which describes the sets of words (and their frequencies) of each document;
  - Normalize the vectors;

  - For each term occurring in each document, replace the term by its hypernym(s). Reweight the corresponding frequency by the hypernym level (hypernyms close to the original terms should be weighted heavier than a less specific high category) and a ration relevant to the number of hypernyms that is generated from each term (to ensure the resulting weight per term is still proportional to its importance in the document and not to the number of hypernyms it has);
  - Compare the resulting categories to the original WordNet categories and determine of some entire categories can possibly be discarded.

- **Result:**

  The method was implemented, however the resulting vectors were so large and vague that no clear conclusion could be made from it. The main reason for this is the ambiguity of terms. For example, the term "bank" has so many meanings to multiple groups.

- **Discussion:** Insufficient results. Research aborted.

5.4.11 Determining Black Listing

Documents with unappropriate (unsafe, unfit or harmful) content, as specified in the requirements analysis document. We determine a set of non-relevant documents by documents retrieved by classic search engines using query topics from the black list. Furthermore, all related concepts in those documents retrieved using black-listing terms are also likely to belong to the black list. All documents linked to a black-listed document also belongs on the black list.

- **Hypothesis:** Documents containing a high ration of black terms are black documents.
- **Goal:** Determine a list of black terms and a method to rate documents accordingly.
- **Approach:**
  - Using the list of unfavorable topics described in section 3, use WordNet to expand that list with related terms (synonyms, hyponyms, meronyms).
  - For each document count the number of terms that coincide with the blacklist, and normalize over the length of the document. The resulting value is the score between 0 and 1 (0 is white, 1 is black).
  - By hand select several unfavorable and favorable documents and test to determine a threshold value for the score with which we can determine a document to be black.
- **Result:** The result is a list of black listed terms. However, due to disambiguities in language, a term in the black list may be used in another context. Take for example the word "joint". It can refer to drugs or human anatomy, obviously one context is favorable and the other isn't. As such we have decided that the score is determined by the number of black concepts normalized over the length of the document. Only documents containing relatively many black words (in proportion to the document length) will be classified as black.

  Initial testing has shown white documents to yield a score of 0 rounded off to three decimal points. News articles having some negative words (bomb, terrorism, drugs,

Topic	Evaluation
Query-Document Relevance	✓
Reweighting for Collection	✓
User Relevance Feedback	✓
Query Concepts	✓
Document Concepts	✓
Directory Description	✓
Stemming Document and Query Contents	✓
Self-learning Relevance Score	✓
Terms of Childrens Documents	X
Terms Categories of Childrens Documents	X
Determining Black Listing	✓

Table 5.1: Evaluation of IR investigation results.

etc.) generally have a score of 0.2. Documents that are clearly black (only about illegal behavior) attain a score of 0.4 or higher. The threshold was set to 0.4.

- **Discussion:** Preliminary results showed to be positive. However, more testing should take place to determine the accuracy of the threshold.

5.5 Results

Table 5.1 summarizes the results of the information retrieval research, indicating which aspects have been subject to investigation, and whether initial results indicate that it is useful in the context of this project or the field of Information Retrieval as a whole.

Though children's search engines are in abundance, there are certain aspects which we researched here that may be of aid. For example, determination of documents relevant to children does not yet occur automatically, neither does categorization. Because this occurs manually, the contents are not up to date. Automatic retrieval will assure a much more timely update of the repository, and is furthermore not liable to errors due to human judgement or bias. If tested more adequately, possibly our results may be of use.

5.6 Conclusion

The goal of the information retrieval research was to identify methods or procedures that will aid in the goal of project, to successfully design and implement an easy to use tool that supports children in their search for information on the Internet.

We had broken this down into two subgoals (see section 5.2.1). The first has been investigated, tested, examined and evaluated in chapter 4 and the discussion will not be repeated here. For the second we completed a literary research of relevant existing research. From that we alleviated certain algorithms and methods which could possibly aid us in our goal. We implemented those that seemed relevant to our goal and tested their results using our prototype. The underlying hypothesis, goal, method and results have been stated in section

5.4, in each case followed by a conclusion which indicates whether it may be useful at all to continue investigation and testing in this field or whether our hypothesis was completely on the wrong track and we see no future in elaboration. Whether the preliminary results were positive or negative has been summarized in section 5.5.

5.6.1 Evaluation

The literary research really helped us in the right direction and it was fun to understand and implement the algorithms, which has really been a learning experience. It was also an eye opener to understand how other search engines are currently retrieving their documents, and how rather simple algorithms achieve high performance. Though exact algorithms being used by well-known search engines are hard to find in literature, most state of the art search engines currently only make use of the vector-space model (with exception of Google's PageRank). We were able to implement the same model which was very exiting.

We were obviously unable to research and investigate all existing research in the field of Information Retrieval. However, by means of following references of published scientific articles, we were able to determine which algorithms were well-known and well-tested and the results published, and thereby assume their quality in terms of validity, reproducibility and precision.

Chapter 6

Implementation

6.1 Executive Summary

The design models (see chapter 4) form the basis for the implementation of the requirements (described in section 3.5.1). In some cases alternative methods or procedures have been chosen. This chapter discusses the implementation decisions (i.e. which algorithms have been chosen, their implications, assumptions and prerequisites) made while coding the product (see section 6.3). The resulting code is verified against the requirements in section 6.4. Complete testing, including validation, verification and evaluation, of the result is described in the next chapter, chapter 7.

6.2 Introduction

As with anything one does, during execution of a plan you run into problems, realize that your plan is merely impractical in some aspects, or realize that there may be alternative simpler manners to do things, as so did we! The component distinction made in section 4.5.2 showed to be everything but flawless. The descriptions and the responsibilities were well thought out and adequate, yet to contain several functions in particular components showed not to be absolutely practical. As such, the functionalities remained and no high-impact adjustments were made, yet some shifting of functions between components has taken place. Because the waterfall model is an iterative process, changes in the design made at the time of implementation should be adjusted in the design models.

Because of the amount of time and work it costs to produce and approve documentation, iterations involve significant rework. Therefor, after a small number of iterations, it is normal to freeze parts of the development, and continue with the later development stages [26]. The result is that the code no longer distinctly reflects the design, yet care is taken to ensure that any modifications yield the same result and characteristics for the program as a whole, and thus still adhere to the requirements posed in the requirements analysis.

6.2.1 Goal of Implementation

The goal of implementation is to generate program code that adheres to the established specification, in order to ensure successfully implementation of an easy to use tool that supports children in their search for information on the Internet.

The formal design models described in chapter 4 form the input for this stage. Due to their nature, these models describe the program to be built with such detail that one-to-one translation can take place into code. In some cases we deviate from the model and implementation decisions had to be made, possibly because the model was incorrect, possibly because during coding an other alternative seemed simpler. However, when this is the case, the design is reviewed. Yet, due to the amount of rework this involves, at a certain stage the design is reviewed but the design models aren't. Those modifications are documented and justified in this chapter.

The requirements formed the specification of the product. These were translated into a design which described the specification. The validity of this step has been justified in chapter 4. This was in turn translated into code. The validity and quality of this step becomes evident during testing. Yet, due to the nature of the formal designs that form the basis for coding, a one-to-one translation leads to an accurate implementation of the specification. Problems may arise when, during coding, one deviates from the original design. By performing unit testing and mental verification we try to alleviate any problems. Anything we may overlook will come to light during the test phase, during which this chapter can form as a basis to pinpoint any possible errors for debugging. The result is a program which adheres to the (primary requirements) of original specification that has been established during the Requirements Analysis phase.

This chapter describes exactly on which points the code deviates from the original design, and why, and justifies the implementation decisions made. This chapter has been written with several goals in mind. The first is to show our supervisors that we had to deviate from our original plan, yet are capable of justifying these reasons, and upon running into problems, we are capable of coming up with a new solution to deal with such a problem, yet yielding a program that has the same behavioral characteristics that were set by the specification. This chapter furthermore communicates our ideas to any person who is working on a similar project, or continuing with our project, to indicate on which points our resulting deviates from the original design, why, and what the alternative path is. This can be useful to avoid falling into the same pitfalls, yet also useful to understand the current design in the case that the current program is to be elaborated on. Furthermore, in the case that errors are encountered during testing, we have indicated that this is most probably due to implementation decisions which have been made that deviate from the original design. This document will help pinpoint the potential source of the errors because it describes the location of these deviations.

6.2.2 Implementation Process

The design incorporated all requirements, primary and secondary. To create a program that adheres to the project goal, it is only mandatory that the primary requirements are implemented. With these requirements, the program will exhibit all minimum required characteristics and behavior. It therefor seems natural to initiate with the implementation of the primary requirements. We initiated implementation with establishing a framework that was required to run the system (e.g. setting up, connecting and communicating with the database), a primary requirement. Implementation is an iterative process in which primary requirements are dealt with and coded one by one.

To reduce complexity, implementation is broken down into smaller steps, by breaking down the system that is to be built into smaller units. See figure 6.1 for a graphical representation

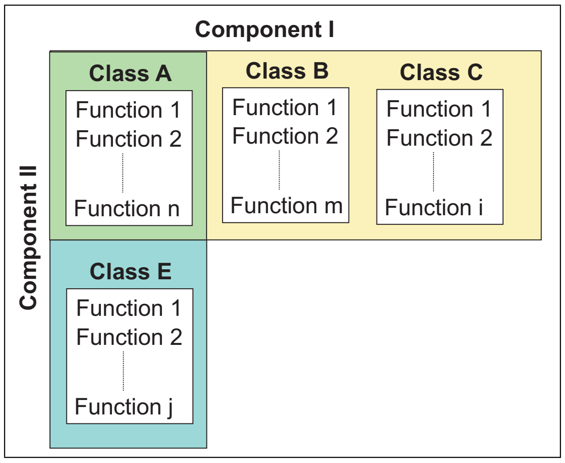


Figure 6.1: Graphical description of a system.

of a system and its subunits.

The process of implementation starts with class implementation. A class is created containing one or more function (collaborating functions are called units). Each function and unit programmed is individually tested directly after implementation, to ensure that it does not contain errors and adheres to its specification, and validated that it still exhibits the behavior that is expected from it (even though it may deviate from the original design). Whenever a function or unit is modified, unit testing is carried out again.

Testing on unit level can dramatically shorten and simplify an otherwise complex testing process because the source of an error is restricted to that class domain and can therefor be pinpointed easier, also simplifying the verification that the error has been solved because overview can be maintained. Though certain levels of testing are interleaved with the implementation process, it is mentioned here because it plays a distinct and important role during the implementation phase. An error on a class level will remain erroneous on the package or system level, troubling implementation. However, testing is a complex process and there are many levels of testing. A separate chapter has been devoted to testing (see section 7.3.2) which describes testing procedures, justification of completeness as well as the results of testing.

Separate classes can be integrated together, borrowing functions from one another. A set of collaborating classes is called a package (or component). Whether this integration process is flawless is tested during integration testing (the results are documented in 7.3.2). Similar to the integration of classes, packages can be integrated to collaborate together to create a system. The system as a whole is to have all the functionalities described in the specification. The resulting system is tested for errors during system testing (see section 7.3.2) and tested for behavior and functionality during validation (see section 7.4).

6.3 Implementation Decisions

This chapter discusses which decisions and assumptions we have made in line with the implementation of Giggle.

6.3.1 Class Implementation

On the level of class implementation we generally speak of functions and algorithms which have been implemented to fulfill a particular functionality. These functions are bounded to their own class, the so called `private` functions.

In the case that standard well-tested and well-documented algorithms have been implemented in the system, these are discussed in this chapter. In the case that additional parameter tuning, algorithm adaption, or specifically an alternative or new strategy has been chosen, these strategies and their results will be discussed in the section 5.4.

- **Active Search State**
  In order to ensure that between different interfaces and pages the user does not lose his or her current result, we store variables in the database which reflect the current search state, such as documents in the results list, the query input, and the history path. This is to refill a screen after refreshing.

- **Analyzer**
  For purposes of performance of both speed and storage space, documents are analyzed for their content which is translated and internally stored in a vector representation, rather than maintaining the entire original document. Most common is storage in a term-frequency vector which sums up how often each term occurs in a document. One must note that the order of the terms in the document is lost in this process. Thus, each document is transformed into a vector V such that v_i is the tf_idf weight of term t_i. Though we undoubtedly loose some precision in describing the document, storage of a document in vector form allows for simple manipulation in the form of normalization, and a simple and fast method of comparison to other vectors (and thus queries or other documents). To elaborate further, we choose to store not only terms but also term-pairs, every combination of adjacent terms in the document. A subset of these pairs contains concepts which can be used to increase retrieval precision.

  The TF-IDF term reweighting technique attempts to find distinguishing features in documents by making words occurring in a document frequently more important (tf: term-frequency), and makes rare words across a collection of documents more important (idf: inverse document frequency).

- **Database**
  Storing a copy of web pages locally is too expensive, however accessing remote pages through the network at query time is too slow. To improve performance, inverted files are used for indices [106], this means that for each concept a list of relevant URL-links is stored.

- **Directory**
  It is impossible, and explicitly not even wanted, to show every concept in a directory structure. The main reason for this is that a directory is to give overview and structure, and alleviate the user by only allowing a small number of choices. A history option is given for easy reversal of actions in the case the user makes an incorrect selection.

  A directory structure has been proposed by comparing existing directory listings (for adults and those specifically for children) with features and constraints (such as cognitive aspects) that came to light during the Requirements Analysis. Initially the documents from this children's search engine have been used to fill and train the system. After training, the system is capable of determining to which directory a new document belongs. To determine the documents and topics to be withheld within each category, the prominent terms (and the supercategories they belong to) in the documents pertaining to such a directory are to be listed and used to automatically determine the category given a document.

Though the requirements indicate that a selection in the relationship-map is to alter the directory, this has shown to be impossible. Because terms can belong to multiple categories, visualization becomes a problem. We have had to drop that requirement.

- **Document Preprocessing**
  During preprocessing of a document, its content is stripped, stemmed, placed in a vector, and normalized for efficiency purposes (See item Lexicon below for a more detailed description). Furthermore, the document content is analyzed for prominent concepts, determined if blacklisting required, determined to which directory category it belongs, and indexed for timely retrieval.

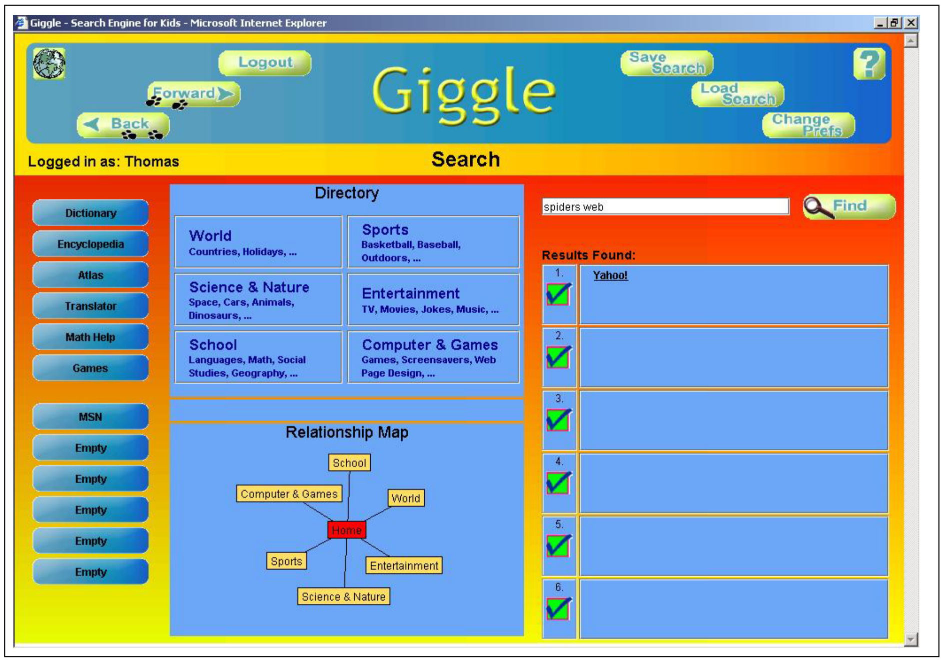
- **Help**
  There are two important aspects to the help. The technical aspect which shows the help on the screen, and the content of the help. The help must be reachable from each other interface page, every page must have an icon (typically a question mark) linking to the help page.

  The content of the help page must be composed of an introduction (what is Giggle and what is it for) a list of options (which options are there and what are they for), and directions on how to use those options (including screenshots with arrows pointing to buttons for clarity). Also, the descriptions must be written in simple, clear and concise language, easy for a child to understand. Furthermore, the interface must be attractive and colorful to make it less boring (using help pages are usually not fun at all!) and to indicate that children are the primary target group for the help pages. Instructions from online children's games are used for a basis. Afterwards, children must be consulted to verify that the instructions are indeed clear. It is imperative to ensure that the screenshots included in the help pages and the names of the options are those chosen in the final version of the application.

- **Interface**
  The interface has his own menu and toolbar with buttons such as back, forward, login, save, and load. To prevent confusions the standard task bar of a browser must be removed. This is done using an index page which use a javascript to start the page with the applet in it. The javascript tells the browser to open the giggle site in a frame without the menu and toolbar of the browser himself.

  The main page after login (figure 6.2), the subscription page (figure 6.3), the login page (figure 6.4), and the page used to indicate the users preference (figure 6.5).

- **Language Determiner**
  Several methods for determining the language of a document were researched. According to researcher Kulikowski, using common short words are a very cheap and effective manner to determine the natural language of a document [125]. Thus for the English,



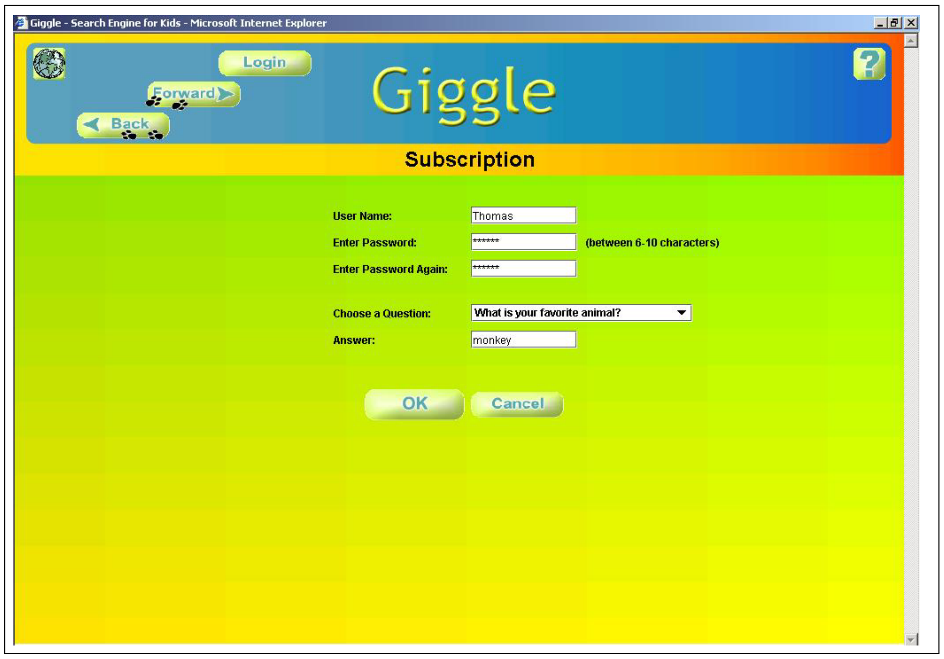
Logged in as: Thomas

spiders web

Results Found:

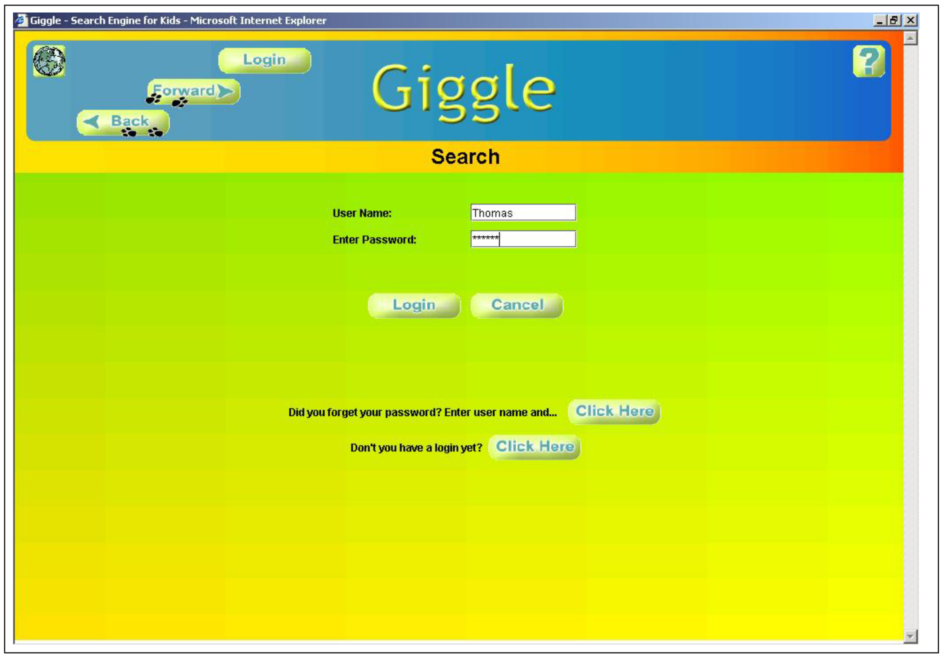
1. Yahoo!
2.
3.
4.
5.
6.

Figure 6.2: Final Interface - Main Page after login


Giggle - Search Engine for Kids - Microsoft Internet Explorer
Login
Forward
Back
Giggle
Subscription
OK
Cancel

User Name:	Thomas	
Enter Password:	******	(between 6-10 characters)
Enter Password Again:	******	
Choose a Question:	What is your favorite animal?	
Answer:	monkey	

Figure 6.3: Final Interface - Subscription Page


Giggle - Search Engine for Kids - Microsoft Internet Explorer
Login
Forward
Back
Giggle
Login
Cancel
Click Here
Click Here

Search

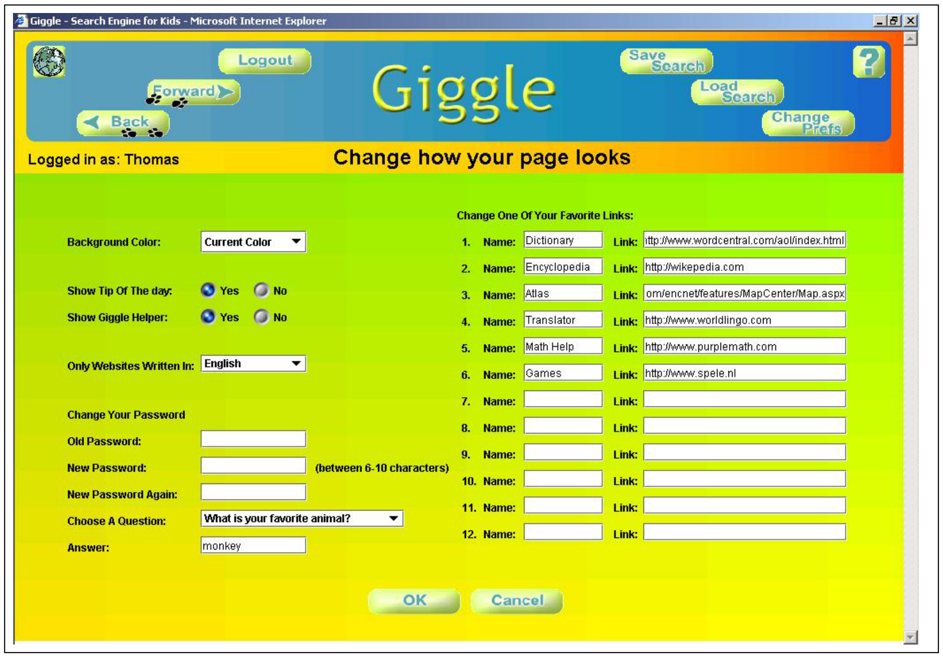
User Name: Thomas

Enter Password: ******

Did you forget your password? Enter user name and...

Don't you have a login yet?

Figure 6.4: Final Interface - Login Page



Logged in as: Thomas

Change how your page looks

Background Color: Current Color

Show Tip Of The day: [✓ Yes] No

Show Giggle Helper: [✓ Yes] No

Only Websites Written In: English

Change Your Password

Old Password:

New Password: (between 6-10 characters)

New Password Again:

Choose A Question: What is your favorite animal?

Answer: monkey

Change One Of Your Favorite Links:

1.	Name:	Dictionary	Link:	ittp://www.wordcentral.com/aol/index.html
2.	Name:	Encyclopedia	Link:	http://wikepedia.com
3.	Name:	Atlas	Link:	om/encnet/features/MapCenter/Map.aspx
4.	Name:	Translator	Link:	http://www.worldlingo.com
5.	Name:	Math Help	Link:	http://www.purplemath.com
6.	Name:	Games	Link:	http://www.spele.nl
7.	Name:		Link:	
8.	Name:		Link:	
9.	Name:		Link:	
10.	Name:		Link:	
11.	Name:		Link:	
12.	Name:		Link:	

Figure 6.5: Final Interface - Change Preferences

Dutch and Spanish languages, the most frequently occurring words (in documents) shorter than or equal to three characters were summed up, about 50 terms per language. Preliminary testing indicates the program is capable of correctly determining the language (English, Spanish, or Dutch) with a certainty of 70%, and yields the value uncertain below this percentage. Furthermore, upon presenting documents in German, Portugees, and French, the program could accurately determine the language. Of the tested languages, only Italian introduced problems because it's discriminating characteristics are similar to those in Spanish. As the database is capable of efficient searching the lists are stored in the DB2 database.

- **Lexicon**
The lexicon deals with all language-related aspects. Traditional grammar classifies words into eight parts of speech; verbs, nouns, pronouns, adjectives, adverbs, prepositions, conjunctions, and interjections. This section describes how the lexicon deals with each one, and why this choice has been made.

  - **Concepts:** An entire research can be dedicated to the process of describing a document with a set of merely 2 or 3 meaningful terms, so called concepts. Though the derivation of concepts and its underlying theories may be complex, they are extremely useful because of their extreme strength in expressive power. Such expressive power may extraordinarily simplify a search process (comparing concepts rather than all the terms in a document separately), increase the feedback to the user and thus the understanding why a document was retrieved, and may furthermore be of much help in classifying documents into directory categories and in the construction of the relationship map with meaningful terms that describe a document.

    We take a step back and have a look at the English language. Nouns are undoubtedly the most meaningful and expressive syntactic category of words. One may argue that a search query is expressed by more than merely nouns, which, our research has shown us is not incorrect. However, stemming any adverb, adjective, verb or noun results in the same stem (e.g. noun: my *neighbor*; verb: the *neighboring* house; adverb: *neighborly* community; adjective: *neighbor* states; are all stemmed to the same term *neighbor*). Though obviously some semantic meaning has been lost during the translation, for search purposes it makes us capable of describing what a query indicates without the burden of dealing with different parts of speech; furthermore, a noun in one sentence can be an verb or adjective in another (i.e. noun: we write in *books*; verb: John *books* the ticket) which would unnecessarily complicate the case even more.

    The most appropriate candidates for determining a concept from a document are the most common terms or phrases (pairs of consecutive terms), because undoubtedly, if the term which occurs most in a document is the term ”bike”, the document is most likely to be about ”bikes”. Prior to determining concepts some steps should be made: stop words must be stripped (an article or conjunction being a concept is inadequate); nouns, verbs, adverbs and adjectives should be stemmed (for reasons described in previous paragraph); synonyms should be considered (to make literature more interesting writers tend to replace terms by synonyms in writing, while referring to the same thing, if possible this should be traced); and normalization should take place with the rest of the collection (if all childrens documents contain the term ”kid”, it has no distinguishing capabilities and should refrain from being a concept).

- **Digits and Numbers:** Numbers on its own are merely too vague to be useful during retrieval. (i.e. a user wanting to know information about all the wars before 1860 will only lead to documents about the year 1860). In general it is wise to discard numbers and strip all digits in terms [106]. Indexing numbers are only useful if the system is willing to transform it to meta-information, that is to say detect whether the number regards a date, a year, financial-value, credit-card number, etc. which can be used to adequately index numbers. Furthermore, some digits like dates can be represented in tens of different forms (9-11, 9/11, 11/9, 9/11/2003, etc.) requiring normalization for it to have any semantic value. However, this is not a primary requirement of the final product and can be implemented at a later time.
- **Stopwords:** Stopwords are stripped from the documents to by the lexicon in order to increase efficiency because their overall contribution is relatively small (they are extremely common in all documents and hold low semantic meaning) and dramatically reduces the indexing structure, however we note that this procedure might lower recall, yet we have decided to implement it because it can compress index size by at least 40% [106]. The most appropriate candidates are articles (i.e. a, an, the), pronouns (i.e. I, you, his), prepositions (i.e. on, over, to), and conjunctions (i.e. and, but, or), which are determined by looking the words up in a stopword-dictionary which we have constructed manually from the above categories (testing has shown that the English-common-word dictionary is inadequate for this purpose).
- **Synonyms:** To increase the effective range of an input query, the terms input can be expanded by their synonyms. As such, a user searching for information about cats will also get information about felines. In essence, though it does not precisely answer the input query, this answers the information need. The underlying assumption is that the children, with their lack of domain and topic knowledge and a low literacy level, will be advantaged by such a query expansion. The expansion of the query vector by synonyms is to be additive to the original query, not replace it, this ensures the user has the feeling to remain over control of the system and that the synonym expansion only aids in the secondary case in which the original term yields no results. Thus the synonym attains a lower weight than the original term.

$$\text{Synonym\_Weight} = \frac{\text{Weight\_Original\_Term}}{2} \tag{6.1}$$

  For each term 0 or more synonyms can be known. We apply synonyms to stems to ensure we expand the query for all syntactical categories of terms, not merely nouns.

  Synonym expansion is also useful in determining the concepts of a document. Reason for this is that writers often replace terms by their synonyms to make literature less monotonous to read. The synonym is replaced for the original term with the notion that is has the same semantic value. Thus, while building up the document vector, upon encountering a term we check if it, or a synonym to it, exists in the vector. All occurrences of the term and its synonyms are increased in value. If the term itself was not yet in the document, it attains the frequency of any one of its synonyms (they should all be the same) and is added to the document vector.

- **Login**
  The search engine can be used without logging in. But if a user log in, several extra features will be available, such as changing the background color, turning on and off of the help and the tip of the day, and storing their personal favorite web sites.

  If a user forgot his/her password, he/she can login at an alternative method. The combination of the user name and the answer to the asked question, which the user has selected and answered during the subscription, can be used for logging in. If there has been logged in in this way, the user is asked to change his/her password so the next time he/she can log in using the password.

  According to the requirements this information should be encrypted, but the effort that has to be put in for encryption is not paying. The information of a user that will be saved, doesn't ask for high protection.

- **Police**
  The requirements posed to the relevance of retrieval results and filtering (see NF3 in section 3.4.2) yielded a list of topics which must be avoided in the contents of the resulting documents. This list has been used to establish a blacklist, terms which seem irrelevant to our user group and should not be yielded during retrieval. Our lexicon has expanded this list with synonyms, meronyms and hyponyms to the original term, and then stemmed the terms to allow for comparison to the internal document representation.

  Merely blacklisting a document if it contains one of these terms is not appropriate; this would lead to inappropriately blacklisting a newspaper article describing a recent bombing.

  As a solution, given a document we count how many blacklisted terms appear in the document, and then normalize over the length of the document. In this case our assumption is that a true black document will contain proportionally more black terms. As such, a document describing how to built a bomb (and thereby describing the bomb and its components) will attain a higher blacklisting score than a newspaper article that merely states a bombing has taken place.

  However, because some terms have multiple meanings, such a blacklist can be problematic. Take for example the term 'joint' which can be referred to from a drugs context (black) as well as a body part (white). To compensate we chose for a probabilistic approach, we check how many different senses or meanings each blacklisted term has, and normalize its weight accordingly. As such, the new score depends on the probability that the term mentioned in the document represents a blacklisted term rather than one of its other senses.

  Upon categorizing a document as black, the Police class sets the base of the URL where it was found in a list. No documents with a URL that can be found in the list is permitted to be stored and used in the database.

- **Query Input**
  Our research has shown that our users are not capable of using boolean operators. We assume, in the case that multiple terms have been input in the query, an AND operator is insinuated.

  Because our document retrieval system makes use of concepts, as does our query input. That is to say, in addition to a query being composed of separate terms, term pairs are also considered.

  Our focus group is prone to spelling mistakes, as such when a term is input that we cannot find in our dictionary, we assume that it has been spelled incorrectly. In this case we find similar terms to the input (by means of letter insertions, deletions, or swapping) and expand the query with these alternatives. The weight given to these alternatives is lower than that of the original term. This ensures that in the case we are wrong and the user did not misspell a term (it is merely not in our dictionary) the top resulting documents will still be relevant to the query input, if there are any. In the case no documents adhere to the query input, the spelling alternatives will be used as a basis for retrieval. See item Spell Check below for more details.

  Our investigation has shown that our focus group has troubles thinking of adequate terms due to a lack of domain knowledge. As such, we expand each term in the query input with its synonyms. A search for 'rasta hair' will be expanded with the synonymous term 'dreadlocks'. Furthermore, terms input in the query are expanded with hyponyms one level higher in the hierarchy. That is to say, both the terms 'bull' and 'cow' are expanded with 'cattle'. For the same reasons as explained above with the spelling alternatives, this new term will be given a lower weight than the original term to ensure the results answer the query and aid only in the case the query is inappropriate. See item Synonyms above for more information.

- **Reading-Level Determiner**
  The Requirements Analysis proposed three methods for determining the reading difficulty of a text, namely (1) The Fry Readability Graph (FRG), (2) SMOG, and (3) The Lexile Framework. All three are based on the same principles, longer sentences and words make a text more complex to read. The Lexile Framework also included word familiarity, the chance of running into a word in another text. Because of this additional aspect, our preference went out to this method. However, we were not able to find any algorithms describing the method, and only a limited number of requests are permitted using existing tools to determine the Lexile level documents, making it unappropriate to our needs. Thus, the FRG and the SMOG have both been implemented, and their average score is taken as the result. Both algorithms require determination of the end of a sentence. Generally this is not much of a problem, however, we ran into problems when it came to tables. Because HTML pages generally have multiple tables embedded within each other, we had to choose to consider each table cell as a separate sentence, which in effect lowers the reading level that is determined for the document. This leads to a lower precision overall, yet a higher recall. Results have been mapped to those of other search engines, such as Ask Jeeves for Kids to test accuracy and compared to the results given by the Lexile website, initial testing results were satisfactory.

- **Relationship-Map**
  Just like the directory, the user can narrow its search query by clicking on one of the

words in the map. The most important word is centered in the middle and is colored red. For determining this center node the most important word (determined by the concept relevance score) from the input query will be taken.

Hypernyms (showing the supercategories of the input/selected term), hyponyms (showing the subcategories of the input/selected term) and meronyms (showing constituents of the selected term) are to be used and presented in a map structure so that users can refine their query. To increase overview, the numbers of terms shown on the screen must be limited to at most fifteen.

Each word type (hypernym, hyponym or meronym) has its own color. This not only gives the interface a colorful look, but by representing word relationships, and it helps users to quickly find the right words to refine their query. The length of edge is determined by the degree of relationship, which is a value between 0 and 1. If, for example, a term is very likely to expand the query, the length of the edge is short.

$$\text{Length} = 100 - (\text{relationship\_degree} \times 100). \tag{6.2}$$

WordNet is extremely large, comprising of a total of 152059 unique terms of which 114648 are nouns and 11306 are verbs. Because in English "every noun can be verbed" and only nouns are used in the directory and relationship map structure, we propose to make the database more compact by removing all verbs, adverbs, and adjectives. We furthermore plan to research the possibility of determining which types of superterms are referred to from within documents for children, possibly some categories are never referenced from within that context and may not have to be indexed.

- **Resulting Document**
  Representing a resulting document was not without problems. The problem was to keep communication between this resulting document and the main program restricting the types of documents which can be shown as few as possible. After all, we use a JEditorPane to show the resulting document. This is an text field which can handle the most common HTML tags, such as frames, images, tables, enumerations, fonts, and applets. But it cannot show more advanced pages written in, for example, flash. However, this is also very positive. As certain types of pages can not be shown, either can pop-ups, which adheres to the user requirements pertaining to user safety.

- **Results List**
  Conforming to the requirements posed to the interface, the requirements list is composed of at most 6 items. Each item is listed in order of relevance (highest relevance on top), with the title emphasized and a short summary including query terms (also emphasized) to give the user both an impression what the document is about as well as feedback why that document was retrieved in relation to the query input (or selection made in the relationship map or directory).

- **Spell Check**
  Terms input by the user are prone to spelling errors. An input term which cannot be found in the dictionary is assumed to be spelled incorrectly. In order to determine relevant variant spellings, Edit distance (Levenshtein distance) can be used. It determines the minimum number of character insertions, deletions, and replacements needed

to make two strings (original user term and alternative spelling) equal. Transposition errors are common in written text, and can be treated as a deletion plus an insertion. Fixing the edit distance to a particular value yields all possibly related terms to a misspelled input term. The Edit distance can be calculated efficiently in $\theta(\mathtt{m,n})$ time (where `m` and `n` are the string lengths) using dynamic programming. For its efficiency reasons, this method has been chosen for the implementation.

Our experimentation has shown us that fixing the edit distance to a value of two character insertions, deletions, and replacements will yield extraordinary amounts of spelling alternatives, in some cases even up to 160 variants. Obviously, the user should not be burdened with having to search through such amounts of spelling variations or results based on such spelling variations. Alternatively, fixing the edit distance to 1 character yields spelling variations between 0 and up to about 10.

Yielding spelling variations can be useful in several situations. The user can be prompted that an input term is likely to have been spelled incorrectly and given the choice to search on an alternative term by just one click of the mouse. Another alternative, requiring no intervention from the user is to expand the user query with the variants yielded by the edit distance algorithm.

We keep in mind that possibly no spelling mistake has been made at all, merely that the word (for example a name) typed in does not exist in our dictionary. Furthermore, upon yielding spelling variations only one of the yielded variants could possibly be correct, and thus from a probabilistic viewpoint, a smaller amount of variations is thus advantageous over many. In order to ensure that the query does not become an ill-representation of what the user wishes to search for, we can weights the additional terms, maintaining an originally input term more important than a yielded term, yet allowing for search results based on a spelling variation. Furthermore, the new weight should be dependent on the number of spelling variations found; the more spelling variations found, the smaller the probability that a correct one has been chosen. For these reasons, we have chosen to weight each yielded spelling alternative according to the following formula:

$$\text{New\_Weight} = \text{Previous\_Weight\_Term} + \frac{1}{\text{Nr\_Spelling\_Variations\_of\_Orig.\_Term} + 1} \tag{6.3}$$

- **Spider**
  For the spider architecture we have chosen for a centralized spider-indexer architecture. A spider is a program that traverses the Internet sending new or updated pages to the main server where they are indexed. Disadvantages of using this method is that the data is quickly outdated due to the highly dynamic nature of the web, large volume of data, and high load at the Web servers. Another method is the distributed spider-indexer architecture which is more efficient yet requires the coordination of several web servers, a task too complex for this project.

  We initiate the spider to start with a set of popular URLs, because we expect that they have information which will be frequently requested. For performance improvement: crawl popular pages more frequently.

  A depth first algorithm is narrow but has a deep traversal, yet a breadth first algorithm has a broader span yet bombards a web server with requests.

- **Stemmer**
  Stemming, as a form of normalization, takes place by the lexicon, mapping all plurals, gerund forms, past tenses and all other syntactical variations of a term to one grammatical root word. Though research results to the benefits of this process are controversial explaining why its wide-spread-use has been inhibited, we believe that stemming will increase retrieval results for our specific user group because of their lack of knowledge of grammar and the functioning of an information retrieval system (i.e. our own research has indicated that children may type in 'cats' when they actually wish to find information about a particular type of 'cat'), and furthermore since a single stem typically corresponds to several full terms, by storing stems instead of terms, compression factors of over fifty percent can be achieved [105]. The advantage of stemming at indexing time rather than at search time is efficiency and index file compression; since index terms are already stemmed, this operation requires no resources at search time, and the index file will be compressed as previously described. Because of its simplicity and elegance, yet yielding comparable results to sophisticated algorithms, a popular suffix removal algorithm is the Porter algorithm [106]. Because the English language knows many irregular forms of verbs, nouns, adjectives and adverbs, prior to applying the Porter algorithms, irregular forms were replaced by their stem using a dictionary (examples are *codices* into *codex* and *are* into *be*). However, we must note that the terms yielded by the Porter stemming algorithm are in most cases no longer real words (*happiness* becomes *happi*); in order to be useful for comparison with dictionaries at a later stage, dictionary entries are also to be stemmed. Problems occur when very short terms are stemmed, such as the term 'ion' which would be reduced to an empty string. For this reason, stemming has been restricted to terms with a length of 4 characters or more.

- **Tagger**
  Because our retrieval system is merely text based, only textual content of a document is extracted and used for retrieval. The tagger matches each input document for html codes. Certain codes may contain useful information such as the title of the document and is stored in the database for feedback over retrieval results to the user. All other html codes are stripped from the document, due to their abundance in every html document they have no discriminatory value. Also, all accents, punctuation and digits are removed from the document because they are not explicitly used for searching, loose all value when the document is seen as a bag of terms, and therefor their storage would merely be inefficient.

  Due to its extraction complexity, we cannot extract terms from objects such as pictures.

- **User Relevance Feedback**
  A simple and well known query expansion technique which incorporates user relevance feedback is the Standard Rochio query expansion. In essence it uses a notion of an ideal query, with an ideal query having maximal similarity to relevant documents and minimum similarity to irrelevant documents. We have created an interface which supports this method. Upon viewing a document, the user is forced to determine whether it is relevant or not, which makes it possible for us to incorporate this information into the original query and use the new query for a new retrieval process which is to be a better reflection of the users information need. So, assuming the user has indicated which retrieved documents are relevant, let D_r be the retrieved relevant documents and D_n be

the retrieved irrelevant documents then:

$$\overrightarrow{q_m} = \alpha \times \overrightarrow{q} + \frac{\beta}{|D_r|} \sum_{\forall d_j \in D_r} \overrightarrow{d_j} - \frac{\gamma}{|D_n|} \sum_{\forall d_j \in D_n} \overrightarrow{d_j} \tag{6.4}$$

The User Relevance Feedback is also used to implement the self-learning aspect of the system. In order to determine which types of documents are generally relevant to the group of children as a whole, it would be unfeasible and incomplete to try to achieve this by hand. This class is responsible for attaching a general relevance score to any concept found in a document. The contents of a document marked as relevant is given a higher relevance score. As such, the system learns which types of concepts are more often desired by children. Upon ranking a document on relevance, the concept ratings are taken into account.

6.3.2 Package Implementation

After implementing each class, and completing unit testing on that class to ensure correct functionality, the classes that belong together are integrated to work as a whole, creating functions that can be borrowing between classes, the so called `public` classes. Problems can arise due to the nature of OO programming, such as inheritance and polymorphism, in which the borrowing of functions from other classes may behavior differently than expected. The expected behavior of each class is depicted in the design of the each package, and chapter 4 has models which describe each package. However, quickly testing if the behavior is as expected is more complex than simple unit testing. Contrary to unit testing on functions, simple logging of variables to the screen dus not suffice as a representation of complex behavior and states.

In the case of erroneous behavior it is most efficient if the bug is detected and dealt with as soon as possible. Not doing so may result in uncontrollably lengthy testing and debugging processes. As such we have chosen to implement on package level in a systematic manner, simplifying the isolation of the source of any error. Starting with an empty package, each corresponding class is integrated into the package one by one, and then globally tested on all borrowed functionalities before the next class is integrated. This process goes on until each class corresponding to a package has been integrated. The same procedure is followed for all separate packages.

6.3.3 System Implementation

System implementation occurs similar to that of package implementation, however, instead of integrating different classes, the different packages are integrated one by one. The main difference is that bugs or unexpected behavior are more visible because the system is to behave as the requirements in section 3 describe, yet isolating the errors is more difficult due to the complex interaction between the different classes and packages. Furthermore, object-oriented programs often do not allow packages to be integrated incrementally because of perpetual dependencies between classes (e.g. class A requiring functionalities from class B which in turn needs functionalities from class C, and so on). Determining if system implementation has been successful is determined during the proces of system testing (documented in section 7.3.2).

Description	**Requirement**	**Result**
Instructions/Help	F1, F3, F9, NF1	✓
Metaphor	F2, NF1, NF2	✓
Query input	F4, F5, F6, F9, NF1	✓
Directory	F3, F4, F5, F9, NF1	✓
Relationship-map	F4, F5, F9	✓
Results list	F7	✓
Result document	F7	✓
Reversal of Actions	F8, NF1	✓
Personalization	NF2, NF4	✓
Links list	NF2	✓
Time and date	NF2	✓
Advanced search	F9	Dropped

Table 6.1: Test results for user-enabling functionalities in product.

6.4 Results

Besides this document, the result of the implementation phase is code. The previous section (section 6.3) has elaborately described which implementation decisions have been made. In the remainder of this section we verify if all primary and secondary requirements have been coded. The subsequent testing phase will test, verify and reflect what the quality of the implementation results are (described in chapter 7).

To ensure that all the primary and secondary requirements (see section 3.5.1) have been coded, tables 6.1 and 6.2 describe what has exactly been implemented or not.

Table 6.1 indicates that the secondary requirement, advanced search, has been dropped during the design phase and therefor has not been implemented. Table 6.2 shows that several requirements have not been implemented. In most cases these are secondary requirements. As discussed during prioritization of the requirements (see section 3.5.1), after the primary requirements have been tested and validated, the remaining secondary requirements will be implemented. At this moment in time they do not yet have to be implemented. They have been included in the list of future recommendation.

However, we also see that some primary requirements pertaining to user safety have not been coded. Filtering banners and advertisements was more difficult than we believed as we have no manner in automatically recognizing the difference between a picture relevant to the document and an advertisement or banner. Though this has been indicated as a primary requirement, the functioning of the system as a whole is not dependent on this feature. To ensure this is dealt with correctly, further investigation must take place, and as we momentarily have limited time, we drop this requirement. This issue has been added to the list of future requirements. Filtering pop-ups has been achieved and implemented.

The requirement of encrypting of user information has also been dropped. The reason for this is that no personal information of the user is stored in the system (the user makes up an artificial user name). After conferring with our client, we have decided that this requirement has become superfluous and choose to drop it.

Description	**Requirement**	**Specification**	**P/S**	**Result**
Query modification	F5, NF3	Spell check	P	✓
		Expansion related terms	P	✓
		Specification category/rel-map	P	✓
		User relevance feedback	P	✓
User safety	NF2, NF3, NF4	Filter popups, banners, ads	P	X
		Filter inappropriate content	P	✓
		Filter on reading level	P	✓
		Warn user leaving site	P	✓
		Encrypt transmission user info	P	*X*
Doc Format Types	NF5	HTML	P	✓
		Microsoft Word	S	*X*
		Microsoft Excel	S	*X*
		PostScript	S	*X*
		Microsoft Powerpoint	S	*X*
Hardware and Software	NF5	Windows 2000 and up OS	P	✓
		IE and Mozilla browsers	P	✓
		Existing filters	P	✓
Performance	NF6	Adequate performace	P	✓
Awareness	B1	-	P	X
Multi-linguistic	B2	English	P	✓
		Dutch	S	*X*
		Spanish	S	*X*

Table 6.2: Test results for supporting functionalities in product.

6.5 Conclusion

The goal of the implementation was to generate program code that adheres to the established specification. As this specification led as the input to the design, the result of translation of the design into code should map the original specification. This was verified in section 6.4. Unit testing was completed on every function and programming unit to ensure that the implementation contained no errors (results have been documented in section 7.3.2). However, we have indicated that in some aspects we have had to deviate from the design. In that case, after unit testing, we verified that the overall behavior was maintained and that the specification was still adhered to. If this is truly the case (thus the implementation actually reflects the specification), can only be determined with integration and system testing during which the behavior of all the components as a whole are examined. Testing is the next phase of the software engineering process and section 7 will describe the quality of the results of the implementation phase. The result of the implementation phase, all the code, forms the input for the testing phase.

6.5.1 Evaluation

A lot of work had been done during the design phase which we profited from. Composing the design models has forced us to think of functionalities and approaches in an early stage. The design model yielded a clear specification of each class, each designed to have distinct roles and responsibilities. This ensured a low coupling between classes, and a high cohesion which simplified programming dramatically because one class could be considered at a time.

Because the two of us were programming simultaneously, it yielded a clear separation between our tasks, ensuring no work was done double and we could work independently without having major integration problems at a later stage. The discrimination between classes also dramatically simplified and clarified unit testing, as it lowered the barrier to documenting testing logs because it seemed to follow naturally. Code components could also be verified for required functionality, as its behavior had previously been described in the design. This gave us reassurance that the code we are delivering is rather qualitative.

However, though the design models clearly indicated which classes were to be built and what they were to do, we did not refine the design model to such a low level that direct translation into code was possible. Creating a model, and especially adjusting existing models, showed to be very time consuming. The primary reason for this is that changing an aspect in one model (let's say, a class diagram) results in a mandatory change in almost all other models (such as state diagrams and sequence diagrams). At a certain point we decided to leave the models as they were. They gave a clear description of which functionalities were required from which classes and how these classes interacted. Because during coding implementation decisions will have to be made, including 'walk-around' programming in which a deviant approach to a function is taken because it is simpler or more efficient to implement than the approach described in the design. Because it is very time consuming to continuously adapt the design model, we agreed that further refinement was to be dealt with and described in comments in the code itself with a description of each class and function. By viewing the design and the code, a person continuing with our project can understand which decisions we have made and why.

Rational is capable of yielding code, under the assumption that the models are complete and error-free. Creating models to such a level is not only extremely time consuming, the code yielded is inefficient, extremely lengthy and very hard to read as it has not been written by a human, but automatically generated by a machine. These aspects yield a code that is hardly maintainable or adjustable. Any future adaptations will require adapting the entire design and generating the code again, requiring the use of the same tool and modelling expertise, restrictions we prefer not to lay on anyone who wishes to continue with our product.

Chapter 7

Testing

7.1 Executive Summary

This section describes the process and results of validation and verification testing of the product built. As a result, we determine not only if the product being built is the correct product in view of our customer and users, but has also been built in a correct and consistent fashion, adhering to design guidelines, to ensure quality. Any problems encountered during testing has been documented in appendix C.1. Section 7.3.2 describes which hardware and software constraints apply to the correct behavior of the system, and thus must be met by the user and our customer. Furthermore, section 8.6 describes which recommendations we have for future elaboration of the product and which must be completed prior to launching the product on the market to ensure the product complies with the expectations of the user.

7.2 Introduction

We could tier the reader by listing the countless numbers of products (and the vast amount of money invested in them) that have been brought onto the market, but never really used because they failed to comply to what they were meant to do. Either the product didn't adequately do what the users wanted it to do (due to an incorrect translation from market demand, to requirements, to design and finally to code), or it was erroneous and not capable of correctly completing its proposed functionalities.

Testing is one of the final processes in software engineering in which we check to see that the product has not only been built correctly, but also if the correct product is being built. This is a mandatory stage prior to launching the product onto the market, because an unacceptable product can lead to a great loss of money, effort and reputation. Unhappy customers will unquestionably lead to complaints and products being sent back, frustrating all those involved such as programmers, marketers, CIOs and other company staff. Software testing is a principal means of avoiding such defects and resultant recalls. It is for this reason that the validation of software, using the principles and tasks listed in section 7.2.2, has been conducted in many segments of the software industry for well over 20 years [24]. It is imperative, to ensure a successful product, that aspects such as maintainability, dependability, efficiency and usability are considered and tested.

In terms of quality management, companies in the business of developing, manufacturing, and marketing software often reflect on ISO standards to determine how good a product is

[26] [27]. However, testing quality goes further than the determination of errors in code. The ISO 8402:1994, treats 'verification' and 'validation' as separate and distinct processes [28]. Verification and validation are processes that determine whether the software conforms to its specification (described in section 3, the Requirements Analysis) and thus meets the needs of the customers.

7.2.1 Goal of Testing

The goal of testing is to verify and validate the result of implementation of the project goal, to successfully design and implement an easy to use tool that supports children in their search for information on the Internet.

The result of implementation, the prototype, forms the input for testing. To complete testing in a structured manner, the goal has been divided into smaller, more manageable subgoals which, when completed, complete the test phase. Thus, during this phase we intend to validate that the program does what the user requires, and verify that the program meets its specification.

The goal of the verification and validation process is to establish confidence that the software system is good enough for its intended use, or 'fit for purpose', and thus test the system on all fronts rather than merely erroneous code. Thus, validation is intended to show that the program does what the user requires, and verification is intended to show that a program meets its specification.

This chapter describes the approach to be taken in testing the software tool that has been built. The results of the test phase describe which of the requirements have and havent been adhered to in the prototype. This summary is in turn used to determine whether or not the tool has successfully been designed and implemented. If all tests pass, the tool is said to be successful. If not, the tool is said to be completely or partially unsuccessful. All errors encountered are logged immediately, and prioritized according to the crucialness of its effects. Crucial errors are fixed immediately, however, due to a time restriction, not all errors can be fixed. Characteristics which have not been implemented (or designed) correctly are logged and form a basis for recommendations for future elaboration of the product.

This chapter has primarily been written for the contract owners (us) to clearly establish, identify, and document product errors and accomplishments. However, it will also be used between us and the client (IBM) as a manner of communication, to establish correct expectations about what has and has not been accomplished throughout the project and to identify issues that must be dealt with prior to final launch of the product.

7.2.2 Testing Process

"Test early, test often, test enough"[27]. The evaluation work created during the software development process is done throughout every stage of the development process. Whether a design model or a procedure in a code, immediate reflection and evaluation takes place in order to uncover possible defects.

We give a summary of how we have incorporated testing in all phases of the project process up to this point. We talked to stakeholders to make compromises about and get feedback on the requirements we have found (Section 3). We worked with users to get feedback on our use-case diagram (Figure 4.2) which formed a basis for all our design models (Section 4). We created an interface prototype and together with users walked through the different use cases in order to evaluate it and pinpoint deficiencies (Section 4.7) in the solution as a whole, as well as collecting feedback with a sample interface (shown in section 4.6). At all stages in the project we sought feedback from users, asking their opinions and watching them work with existing tools and ideas we wished to incorporate in our tool. The investigations with respect to Information Retrieval were implemented in the prototype and thoroughly tested for practical uses (tests are described in section 5.4). An other form of testing is that we sought permission and constructively advice from all our supervisors (at IBM and the university) prior to continuing to a next stage in the software engineering process. The following step in the software engineering process was coding, to which the remainder of this chapter has been dedicated to.

Testing object-oriented software is different from testing procedural software. On one hand, object-oriented programming language features of inheritance and polymorphism present new technical challenges during testing. On the other hand, the OO analysis and design models described in section 4 are similar in structure to the content of the resultant OO program; allowing testing to begin with the review of these models (described in section 4). In order to reduce complexity and efficiently and adequately pinpoint bugs and errors, the classical strategy for testing computer software begins with testing the smallest functions and incrementally increasing the code tested until the entire system has been tested once code has been generated. Otherwise known as unit testing, this process is completed directly after creation of a function or completion of a unit. Modification of a function or unit requires it to be tested again. A series of tests are designed that check class operations and determine whether errors exists as one class collaborate with other classes. As classes are integrated to form a larger system, testing is applied to the whole to exercise collaborating classes. Finally, use-cases are used to uncover errors at the software validation level.

We initiate the test phase by completing verification of the code. As such, we begin with unit testing, then progress toward integration testing and system testing which yield the system's hardware and software requirements, all of which are described in section 7.3.2. The process is completed with validation of the system (section 7.4). With these tests, the functional and non-functional requirements described in table 3.4 are tested. For a reflection on usability evaluation, we refer to section 4.7. For a reflection on the business requirement, we refer to section 8.

7.3 Verification

Verification is the process in which one checks whether the product is being built correctly. This low level activity involves reviewing the work steps and deliverables during a project to ensure they are acceptable. In order to break down the complexity of verification, we break down the process into two subcomponents:

1. Quality: Determining if the system is consistent, adheres to standards, uses reliable techniques and prudent practices.

2. Functionality: Determining if the system performs the selected functions in the correct manner.

The first aspect is done by verifying the process and product results according to a checklist based on IBM's best practices (section 3.5.2 and appendix A.2.4). This will be addressed in section 7.3.1. The second is accomplished by comparing the resulting program to the design models described in section 4. Because the classes in OO programming represent the classes used our modelling methods, this process is much simpler than verification of hierarchical (top-down or bottom-up) programs. This verification aspect is addressed in section 7.3.2.

7.3.1 Quality

During the Requirements Analysis phase we investigated and summarized IBM's testing methods with respect to desirable outcomes of software processes and products. We use this list of characteristics and guidelines (see section A.2.4 and IBM's AMS NL software checklist [10]) as a checklist during the verification proces.

Adhering to the software design checklist ensures a correct and qualitative description and translation of requirements into design and the implementation that follows, and a systematic approach to implementation ensuring a robust product. Specifically, the following criteria have been specified during the Requirements Analysis phase and are now evaluated by indicating (P) Pass or (F) Fail followed by a short indication where this can be validated.

- $\surd$ - All requirements must be understood - Section 3;
- $\surd$ - Human factors must be addressed - Sections 3, 4.7;
- $\surd$ - Each new function must have an accurate and complete description - code;
- $\surd$ - There must be mapping back to Software Requirement Specification (traceability) - Section 4;
- $\surd$ - The design must be feasible - Section 4;
- $\surd$ - Product must consist of a reasonable module breakdown - Section 4;
- $\surd$ - Interaction between database and application should be maintained by only one module, that way, changes made in database (e.g. commands) can be dealt with by adaptation of merely one model - Section 4, code;
- $\surd$ - Data files must be clearly specified - code;
- $\surd$ - Database definition must be clear - Section 4;
- $\surd$ - Provisions for database expansion must be made - Section B.3;
- $\surd$ - Provisions for logging errors must be made - Section C.1;
- X - Performance factors must be estimated - does not apply to prototype, section 3.5.1;
- $\surd$ - Database fields must be accurately defined - Sections 4, B.3;
- $\surd$ - Valid error conditions must be specified - Section B.3;
- $\surd$ - Error messages must be defined - code (Database, Interface);
- $\surd$ - Dependencies must be identified - Sections 4.4, 4.5, B.3;
- $\surd$ - Internal interfaces must be defined - code (Database class);

- $\surd$ - Logic errors or processing omissions, particularly in the handling of exceptional conditions, must be dealt with - code, Section testing 7;
- $\surd$ - The design must provide an appropriate level of detail to support the detail design procedure - Sections Application model 4.3, Analysis Model 4.4, Architecture model 4.5, Database design B.3;
- $\surd$ - The modules must be designed to be reusable - design, Section code 4, tested OO classes low coupling, high cohesion, abstraction;
- $\surd$ - Where applicable, existing systems should be reused - code (Database, Interface) 4;
- $\surd$ - The module must be modifiable - design, Section code 4, low coupling, high cohesion described in section 4;
- $\surd$ - AMS NL methodology and standards should be adhered to - Sections Design 4, Coding 6, Testing 7;
- $\surd$ - The project's standards should be adhered to - Sections Requirements 3, Design 4;
- $\surd$ - Data encapsulation must be performed, as appropriate - code (Java classes) and section 6;
- $\surd$ - If appropriate, design language statements should be used - code (existing statements in Java 2 Platform, Standard Edition) and section 6;
- $\surd$ - The design should be made reusable for other systems - design models, section 4;
- $\surd$ - Mental verification must be performed for zero defects - design models section 4, and section 7 testing;

7.3.2 Functionality

This section describes the process of verifying the system's functionalities. The design and implementation phases were interleaved using the waterfall method. That is to say, the product to be built was broken down into several modules and components during the design phase. Each component was designed, implemented and tested individually. After completing the tests successfully, the components were combined and the entire product was tested for functionality and requirements fulfillment.

In order to ensure that testing and bug fixing occur systematically and that no portions of the code are ignored during testing, we have chosen to set up a list of all components and classes that must be tested and log all test results accordingly. Upon testing, major bugs or deficiencies have been noted in Appendix C.1, including an indication of how critical it is to fix them immediately. Upon fixing, this is also documented. The priority assigned to errors is dependent on the its effect on the entire system to function correctly; crucial functions are given a high priority. All errors marked having a high priority must be dealt with immediately before progressing to ensure correct functionality of the system. Such a log is useful not only to structure testing and bug fixing for the programmers, but also for the client to get an impression of the quality of the software and any programmer who is willing to carry on with the project in the future. The log can thus be used to interpret the overal quality of the system.

Unit Testing

Each unit, class, or instance of a class (object) in object-oriented software packages attributes (data) and the operations (methods) that manipulate these data. The smallest testable unit is thus an encapsulated class or object. Equivalent to unit testing in traditional software is class testing for object-oriented software.

Unit-tests are a good way to flush out certain types of defects. Since unit-tests only run on one subroutine they are easier to use, faster to build and run, allow more comprehensive testing on a wider range of input data, help document how to use and check for valid answers, and allows faster testing of individual pieces.

Unit, or class, testing is accomplished by testing all functions according to their expected behaviors as described in section 4.5.2. For each component we tested if the expected behavior was the result after expected input. More specifically, by analyzing the code the following aspects were tested:

1. Applicability of requirements described in section 4.5;
2. Every line of code is executed;
3. Check that the full range of possible input data works;
4. Boundary analysis, thresholds of logical statements are correct;
5. Check for bad input data;
6. Test for scientific validity, review states where the answer is known analytically.

The results have been summarized below and logs in Appendix C.1. Note that this is a summary, of the above aspects, only those yielding unexpected or extraordinary results are mentioned, otherwise the reader may assume 'no new is good news' and the test case is passed.

- **Active Search State** Passed
  The active search state stores all variables which describe the current search state, such as the input query, modified query, results list and indicated relevance. Upon loading a search state these variables are set to the interface. Active Search State has passed unit testing.

- **Analyzer** Passed
  The analyzer adequately translates stemmed documents and query to a normalized term-frequency vector. Terms and concepts may be no longer than 200 characters in length. Relevance between documents and queries are determined adequately with no relevance indicated if the case that no concepts or keywords of distinguishing value (frequently occurring terms are omitted) overlap. Frequencies must be a positive value. Analyzer passed all unit tests.

- **Database** Passed
  To enforce low coupling, the Database class is the only class that can and may communicate with the database. Each function explicitly indicates its paraments and return value and has been tested to justify its truth. All database errors are caught and essential SQL errors yielded and printed to the screen when necessary. Due to internal DB2

logging, errors while inputting document content into a table is automatically accounted for. Database passed all unit tests.

- **Directory** Passed
  The directory structure has adequately been determined and filled through means of training. Navigation through the directory is possible, as is reversal of navigation actions by means of the history search path. The Directory has passed unit testing.

- **Document Preprocessing** Passed
  The Document Preprocessing class adequately yields a term-frequency vector representing the stemmed and stripped content of an input document. The Document Preprocessing class has passed unit testing.

- **Help** Passed
  All functionalities available on all screens of the interface have been described in the help. Also included are a general description of Giggle, a FAQ, and some special tips on improving search results. Users have verified that the instructions are adequate and clear. Furthermore, the help can be reached from all interface screens. The Help passed unit testing.

- **Interface** Passed
  The tested interface from section 4.6 has been implemented entirely according to the design description. The appearance of the interface has changed slightly in relation to the design in light of the feedback our users gave us on the prototype, yet the functionality remains unaltered. The interface passed unit testing.

- **Language Determiner** Passed
  The Language Determiner is able to determine the language of plain text documents of English, Spanish or Dutch documents with a threshold certainty factor of 70%. Upon obtaining an uncertainty greater than 30% the program returns "Uncertain". Given an input document written in another language, it most often makes the correct interpretation. However, some languages have been found to be problematic. For example, German and Dutch, and Spanish and Portugese lay so close together linguistically that the system is not capable of differentiating between the two.

  Furthermore, the system is not capable of making correct assumptions on very small documents. Documents must generally be of length of at least 100 words and variable contents (at least 25 different words) to make a chance at correct language detection.

  Extending the Language Determiner to deal correctly with other languages than merely English, is a secondary requirement and thus does not account for a requirement to determine failure. The Language Determiner passed unit testing.

- **Lexicon** Passed
  The Lexicon class accurately strips accents and digits from a given input. It furthermore is capable of determining concepts, synonyms, meronyms, hypernyms and meronyms and ensuring that such is stored in the database. The Lexicon passed unit testing.

- **Login** Passed
  An unregistered user can register by giving a user id, a password and the answer to a question in order to authenticate themselves in the case they loose their password. A

registered user can login user a correct combination of a user id and a password. An incorrect combination will not authorize login. A user that forgets their password can recover it and is permitted to enter a new password if the user answers the question correctly. An incorrect answer will not authorize login. The Login passed unit testing.

- **Police** Passed
  The Police is able to distinguish inappropriate documents from appropriate documents, according to the document content description stated in NF-3 of section 3.4.2. We collected a set of documents and manually labelled them as *black* (inappropriate documents for which there was no doubt they must be filtered), *gray* (unfavorable documents containing inappropriate terms but for which the content was negotiable) and *white* (favorable documents with no distinct doubt that they are appropriate for our focus group). The Police gives a document a rating, with 0 begin appropriate, and 1 being most inappropriate. All documents in the set *white* were scored with 0. All documents in the set *black* were scored with a value between 0.3 and 0.6. All documents in the remaining set *gray* attained a score between 0.1 and 0.4. Based on the results a threshold value of 0.2 was set to *white* and *gray* documents, all documents scoring below that value are permitted for retrieval, all others are not permitted for retrieval. A threshold value of 0.4 was set to *black* documents. For these documents, the internet domain from which they were retrieved is placed in a blacklist, all documents from related pages will be blocked. The Police class functions adequately and passed all tests.

- **Query Input** Passed
  The query input is correctly parsed, stripped, stemmed and placed in a term-frequency vector. Spell check yields known spelling variations to unknown terms and expands the query, yet with a smaller weight than the original term. For all known terms synonyms and hypernyms are added to the query vector, with a smaller weight than the term that generated them. Query input passed unit testing.

- **Reading-Level Determiner** Passed
  The Reading-Level Determiner is able determine the reading level of a document. Both the SMOG and the FRG algorithm had been implemented, and a function written to determine their average as the final reading age. However, the FRG algorithm showed to be exceptionally slow. We were thus forced to drop it and only make use of the SMOG algorithm. Generally the algorithm works well, adequately being able to distinguish between content difficulties. However, the advised age level is much higher than expected. We thus use the reading-level determiner merely for guidance. Documents closer to the correct reading age will be more appropriate and therefor gain a higher relevance score. Yet documents which attain a high score will not be filtered, but attain a lower relevance score. In effect, these will only be retrieved if no document of attractive score exists in the database. All units tests passed for the Reading-Level Determiner.

- **Relationship-Map** Passed
  Initially a reflection of the directory, the relationship map shows a central term and related (higher or lower in hierarchy) terms around it. Upon selecting a term, the new term becomes the central point and new related terms are calculated. For each term selection, the results list is filled. Unit testing passed for the Relationship-Map.

- **Resulting Document** Passed
  Upon selection of a document from the results list, the resulting document is shown in a new page. The search terms have been highlighted. The user is given the option to indicate relevance of the document, upon which the user is brought back to the original search page. Unit testing for Resulting Document passed.

- **Results List** Passed
  The Results List gets a list of documents from the analyzer. For the top 6 documents in the relevant documents list, a selection of the text is constructed of merely a few lines containing some or all of the input query terms highlighted. The summary and document title of those top 6 documents are shown to screen in a clickable box. The Results List has passed unit testing.

- **Spell Check** Passed
  The Spell Check works adequately on English documents. Our system makes use of a dictionary containing approximately 200,000 words. A query term which does not exist in the dictionary is assumed to be spelled incorrectly. Spelling variations (that do appear in the dictionary) very close to the original term are yielded and offered as alternatives. A correct spelling alternative can only yielded when at most one character misplacement, deletion or insertion has been made.

  Because handling input in Spanish and Dutch has been identified as secondary requirement, testing cannot fail on that account. The Spell Check class passed unit testing.

- **Spider** Passed
  The Spider is adequately capable of traversing the Internet in search of new or existing documents. In each case it chooses to update or store new information in the database pertaining to the document. Spider passed unit testing.

- **Stemmer** Passed
  The Stemmer is capable of adequately stemming all terms to their grammatical roots. Testing has shown that only terms longer than 3 characters should be stemmed to ensure accurate results. Furthermore, based on a grand selection of random terms, plural nouns are adequately stemmed to their singular form. All unit tests passed for the Stemmer.

- **Tagger** Passed
  The Tagger is given the contents of an HTML document as a string and adequately parses and tags it, removing all known HTML codes, digits, and punctuation. Letters with accents on them are replaced by a similar letter without an accent. Even on an input with fictional HTML codes, the Tagger behaves adequately.

  Being able to handle other formats than HTML (i.e. XML) is a secondary requirement and thus cannot be responsible for failing this unit. Unit tests passed for the Tagger.

- **User Relevance Feedback** Passed
  Standard Rochio user relevance feedback algorithm was implemented and tested. In some cases negative values appeared in the resulting query vector. Because this was inappropriate, the algorithm had been tuned to yield 0 as the minimal value rather than a negative number. Unit testing has shown that under the assumption that the documents were marked correctly as relevant or irrelevant, the resulting query resembles

Cluster	**Result**	**Classes in Cluster**
Retrieval Functionalities	✓	Analyzer Intelligent Lexicon Police Query Input Resulting Document Results List User Relevance Feedback
Document Preparation	✓	Document Preprocessing Lexicon Reading Level Determiner Tagger
Database	✓	Database Spider
User Personalization	✓	Active Search Login Registered User
Interface	✓	Directory Login Registered User Relationship-Map Resulting Document Results List

Table 7.1: Results of integration testing on clusters.

the user's information need more adequately. User Relevance Feedback passed unit testing.

Integration Testing

The conventional incremental integration approach (integrating operations one at a time in a class) is not possible because of the direct and indirect interactions of the components that make up a class. The form of integration testing that most reflects the functionality of our system is the so called *Cluster Testing Method* [24]. A cluster of collaboration classes (described by the CRC method in section 4.4.5) is exercised by designing test cases that attempt to uncover errors in the collaborations. Because no exact boundary can be made between class functionalities, some classes may appear in multiple clusters. The resulting clusters (see Figure 4.6) and their overal test results are summarized in table 7.1. The test logs, and thus any specific errors that have been encountered during testing can be found in Appendix C.1. Note that overall cluster behavior can be marked as pass, even though an error may still reside in the cluster. This is the case if the error is not high priority and does not necessarily cloud the functionality as a whole. However, these errors must be fixed prior to launching the product on the market.

System Testing

System testing is concerned with testing if the entire system as a whole functions according to its specification. Logs of system testing can be found in Appendix C.1.

- **Performance and Responsiveness**
  Only superficial testing on performance and responsiveness issues has been completed.

  Initial results show that the system is extremely slow. To give a small indication of the current responsiveness and performance we describe the results of a test using the same hardware and software that will be turned over to the client (all other applications have been closed prior to testing). On a collection of 25 documents, inputting the query ”spiders web” results in appropriate stemming and query expansion, and yields 15 documents that contain one or more of the terms in the expanded query. These 15 documents were composed of 14,811 phrases. The retrieval took 1481 seconds (thus 24 minutes and 41 seconds) to complete.

  Some optimization has taken place to increase responsiveness. Following these improvements, the same search took 47 seconds. Though the optimization has led to dramatic improvements, if one wishes to implement the system, much effort should be placed into responsiveness improvement and testing.

  We did not test performance issues such as precision and recall. It is mandatory that this is completed prior to launching the final product onto the market.

  No testing on capacity has taken place. For adequate testing the number of users that can simultaneously make use of the system should be tested. Furthermore, performance measures related to the number of websites in the database should also be tested.

- **Hardware and Software Requirements**

  This section describes the technical and system requirements that our system poses on the hardware it runs. Without complying to these requirements, we cannot assure that the system will run as intended. It is thus mandatory that the client and user comply to these constraints to insure correct functionality of the system.

  The client side is referred to as the hard and software on which the user is working. This must be capable of running the Java Applets designed for Giggle. Minimal system requirements for the client side are:

  - 900 MHz processor (if it is less, the search for the right documents will take longer).
  - 128 MB RAM (if it is less, the search for the right documents will take longer).
  - Internet connection.
  - A browser which can handle java code, for example Internet Explorer or Netscape.
  - Java Runtime Environment at least at 1.3.1 level.

  The system is OS independent and can be run from a browser on any platform.

  The server side is referred to as all the hard and software which IBM must reserve to keep the system running. Server-side includes the software written for Giggle (such as indexer and crawler), in addition to an internet connection to a server on which the webpage can run and a database used to store all the indexed data.

On the server side, the minimal system requirements are:

- Minimum speed and number of processors and the minimum size of the RAM memory depends on the size of the database. Google, for instance, lets different queries run on different procesors and also lets a single query use multiple processors. To handle their workload, Google has thousands of clusters of more than 15,000 commodity-class PCs (a single processor speed range from 533 MHz till 1.4 GHz). In this way, a single query on Google can read hundreds of megabytes of data and consumes tens of billions of CPU cycles in little time.
- Large storage capacity (the bigger, the more documents can be saved).
- The upload and download capacity must be unlimited. If there are restrictions to the maximum size which can be uploaded and/or downloaded, the search engine is of no use. If the spider is bound to the download limit, he cannot keep the database up-to-date or extend the database to get more (useful) information. If there is an upload limit, the chance that the search engine cannot be used until the end of the month is big.
- A Java Runtime Environment at the 1.3.1 level.
- A database. The software is build on DB2 version 8, but with little changes in the code the software can also run on other databases.
- Stored procedures and views on the server. The database software DB2 UDB version 8 on UNIX, Linux, and Windows has the required stored procedures and views. For DB2 on OS/390, the stored procedures must be installed manually.
- The server must be configured for TCP/IP.

There are no restrictions to the operating system, any operating system can be used. But the version of an operating system will restrict the version of the database which can be used.

7.4 Validation

Validation of software, determining if the product being built meets the expectations of the customer, complies with the requirements and performs functions for which it is intended. THis high level activity focusses on the user visible actions and user recognizable outputs from the system as a whole. In order to preform validation we make use of the use cases that have been used as part of the analysis model. The use-case provides a scenario that has a high likelihood of uncovering errors in user interaction requirements.

In this section we discuss the validation of the Giggle software, which is to say we determine to which extent the product being built meets the users expectations, i.e. ”Are we building the right product?”.

We have spent much effort in creating formal models during the Design stage of this project which were used as a basis for implementation. Here too, these models show to be effective. Because the models are a direct translation from specification, and the code is a direct specification to the models, we can use the models to test if the software does as has been modelled. If so, we can acknowledge that the software has been validated. If not, we must return to the drawing board for debugging to solve the problem.

Because validation is user-oriented we start with the extended use-case diagram (Figure 4.2) which provides a scenario that has a high likelihood of uncovering errors in user interaction requirements. This model has also formed the basis for our application model (Section 4.3) in which we describe all user functionalities based on the possible use case scenarios. We walk through the model step by step to ensure that the product behaves as expected. We initiate at step ”User goes to Giggle” because the Requirement Analysis justifies that our user group is capable of doing so.

1. *User Formulates Information Need*: The user must make clear for himself what he is looking for, and able to indicate this. If the user is not capable of doing so, we can offer the capability of browsing (by means of directory or relationship-map) rather than specifically searching. If the user has formulated his information need, at least to a bare minimum, the query input field aids with query expansion and the relationship-map aids with query refinement.
   Result: Validated

2. *Searches by Query*: User types in one or more words in the input field. This query is stripped and expanded accordingly and is used to find relevant documents. The documents are shown in the results list and the relationship-map is updated according to one of the query terms input.
   Result: Validated

3. *Searches by Directory*: User makes a selection from the directory. The directory, the relationship-map, and the field showing history selections made are all updated and results pertaining to the selection are shown in the results-list.
   Result: Validated

4. *Searches by Relationship-map*: User makes a selection from the relationship-map. The selection becomes the center of focus of the map and the field showing history selections is updated. The results list is updated according to relevant documents to the selection made in combination with the last two selections made from the relationship map or directory (if there are any).
   Result: Validated

5. *Gets help*: Upon clicking on the help button a screen with tips and instructions is shown. The help pages can be reached from any other page. These instructions were based on example instructions written for children and have been shown to children who said they were clear. We ensured that every option in Giggle was explained in the help pages in terms of what it was for and how it was to be used.
   Result: Validated

6. *Views Introduction*: Upon going to the site, the user is given the option to get a general introduction to Giggle. The introduction features a general description, explanation how to use each of the features on the site, and tips on how to improve search results.
   Result: Validated

7. *Logs in*: The user clicks on the button ’Log in’, and after authenticating himself using his user name and password, he is shown his personalized page. In the case the user has

forgotten his password, he is also given the option to restore his account by answering a question. A user who does not yet have a login is given the opportunity to create an account after filling in certain questions.
Result: Validated

8. *Personalize site*: The user clicks on the button 'Prefs' and is given all the options he is allowed to personalize on one page. The page offers a help functionality which explains what each option is for and how to use it. Error prevention is applied to password input to reduce the chance of typos to cause errors.
Result: Validated

9. *Saves search*: After a logged in user clicks on the save search button, the system stores all variables which describe the current search state in the database in combination with the user id.
Result: Validated

10. *Restores saved search*: After a logged in user clicks on the load search button, the system restores all variables from the database which describe the current search state in combination the user id of the logged in user onto the screen
Result: Validated

11. *Get list of results*: After query input or a selection from the directory or relationship-map, a results list of at most 6 results is shown in the bottom right of the screen. Each result shows its title and a short selection from the resulting document itself. Upon clicking on one of the results, the document itself will be shown.
Result: Validated

12. *Selects result*: Each result in the results list can be clicked on, just as a button. Upon doing so the result is shown in a separate page with the keywords highlighted. Also a link to the original website is shown. The user can indicate relevance of the document to their information need by choosing the checkmark (relevant) or the trashcan (irrelevant), upon which the user is returned to the original search page which has incorporated the users preferences in the results list.
Result: Validated

7.5 Results

During the testing phase we have tested, verified and validated the resulting product. Table 7.2 gives an overview of which results are the output of which component of the test phase, including an overall score whether the tests as a whole have been passed or not. I remind the reader, as described previously, that the system need not pass every sub-test in order to attain a general approvement.

We once again map the requirements to the current phase, and consider if each of the requirements has passed the test phase, to ensure that none of the requirements have slipped by. Tables 7.3 and 7.4 indicate the results of requirements testing in the code. Where 'Dropped' is indicated in the table, the requirement has not dropped at a prior phase, and we refer to the previous phases for more information. As you can see, all requirements that have formed the input for the test phase have been tested.

Section	Test	General Result
7.3.2	Unit testing	$\surd$
7.3.2	Integration testing	$\surd$
7.3.2	System testing	$\surd$
7.3.2	System requirements	$\surd$
7.4	System validation	$\surd$

Table 7.2: Results of test phase.

Description	Requirement	Test Result
Instructions/Help	F1, F3, F9, NF1	$\surd$
Metaphor	F2, NF1, NF2	$\surd$
Query input	F4, F5, F6, F9, NF1	$\surd$
Directory	F3, F4, F5, F9, NF1	$\surd$
Relationship-map	F4, F5, F9	$\surd$
Results list	F7	$\surd$
Result document	F7	$\surd$
Reversal of Actions	F8, NF1	$\surd$
Personalization	NF2, NF4	$\surd$
Links list	NF2	$\surd$
Time and date	NF2	$\surd$
Advanced search	F9	Dropped

Table 7.3: Test results for user-enabling functionalities in code.

As a future recommendation, we summarize the failures and errors encountered during testing, for which we have not yet had time to fix them, in table 8.6. Because we no longer have the time to fix these failures, to completely adhere to the users expectations these will have to be fixed at a later point in time, and we therefor recommend that they be dealt with. To aid out client in determining how crucial these recommendations are prior to launching the product, we have prioritized the errors according to their impact on the functionality of the product as a whole.

7.6 Conclusion

The goal of the testing process was to verify and validate the result of implementation, and thus to establish confidence that the software system is good enough for its intended use, or 'fit for purpose'. We have thus tested if the design and implementation has led to successfully incorporating all the requirements into an easy to use tool that supports children in their search for information on the Internet.

To achieve our goal we verified that the product was built correctly, and thus adheres to the quality and functional aspects which were expected of the tool (see section 7.3). Furthermore, it was verified that the product built meets it specification and thereby also the expectations of the customer, and thus is an easy to use tool that supports children in their search for information on the Internet (see section 7.4). That the specification maps the expectations

Requirement	**Specification**	**P/S**	**Result**
Query modification (F5, NF3)	Spell check	P	✓
	Expansion related terms	P	✓
	Specification category/rel-map	P	✓
	User relevance feedback	P	✓
User safety (NF2, NF3, NF4)	Filter popups, banners, ads	P	$\pm$
	Filter inappropriate content	P	✓
	Filter on reading level	P	✓
	Warn user leaving site	P	✓
	Encrypt user info during transmission	P	Dropped
Document Format Types (NF5)	HTML	P	✓
	Microsoft Word	S	Dropped
	Microsoft Excel	S	Dropped
	PostScript	S	Dropped
	Microsoft PowerPoint	S	Dropped
Hardware and Software (NF5)	Windows 2000 and up OS	P	✓
	IE and Mozilla browsers	P	✓
	Existing filters	P	✓
Performance (NF6)	Adequate performance	P	✓
Awareness (B1)	-	P	Dropped
Multi-linguistic (B2)	English	P	✓
	Dutch	S	Dropped
	Spanish	S	Dropped

Table 7.4: Test results for supporting functionalities in code. Dropped indicates requirement has been dropped and therefor not subject to testing.

of the customer has been validated and justified after completing the Requirements Analysis phase (see section 3).

Important to note is that section 7.3.2 describes which hardware and software constraints apply to ensure a correct behavior of the system. A user not obeying these constraints cannot be guaranteed a correct functioning system. A user not obeying these constraints cannot be guaranteed a correct functioning system.

During the requirements analysis phase, we defined our tool as a whole to be successful if 80% of the defined requirements to be implemented have actually been implemented, and 70% of our interviewed stakeholders are satisfied with their experience with the tool (see section 3.4.2). We sharpen this requirement slightly; in order to be successful the requirements must not only be implemented, they must pass testing, verification and validation processes. To adequately determine this, we must describe which requirements passed testing, validation and verification, ordered by requirement (rather than type of requirement as has been done in tables 7.3 and 7.4). This has been completed and is shown in tables C.5, C.6, and C.7. The list of 18 requirements can be divided into 48 sub-requirements. One requirement (encryption) showed to be superfluous, and is therefor not considered. Nine other sub-requirements did not pass testing. Thus, 81.25% of the requirements passed testing, and we can therefor say they have been implemented correctly. We have not been able to complete an elaborate test with all the interviewed stakeholders. However, of the people we have shown the tool to (5 children, 2 parents and our client), all interviewees were satisfied. We conclude, that though several components must still be implemented and tested to comply to all the requirements, our tool as a whole is successful.

7.6.1 Evaluation

We have tested the system as far as we have found possible. Due to time constraints not all validation and verification tests have been completed successfully. Especially performance and responsiveness tests have not been executed completely. While building a prototype (as is the current case), functional testing has a higher priority over performance and responsiveness issues. We have adequately documented any problems encountered, which can be solved at a later point in time (see section 8.6). The system may not behave exactly as expected under extreme conditions (such as unexpected input, or abruptly breaking the connection with the database during data transfer), however, test results show it is capable of fulfilling the user's expectations by complying to the desired functionalities.

Chapter 8

Acceptance and Finalization

8.1 Executive Summary

This chapter describes the resulting product turnover to the client, including a list of sub-products (see section 8.3) to ensure that the client is given a product that he can actually use, or elaborate further on. It also evaluates compliancy to the business requirements (see section 8.4.1) as stated in section 3.5.1. It furthermore evaluates the entire underlying research, process, and tool which can be used to determine the overall quality of the product, as well as which aspects yet remain for further product elaboration and research.

8.2 Introduction

Having completed the testing phase, we have entered one of the final phases of the software engineering process, the Acceptance & Finalization phase. During this phase the product will be turned over to the client for acceptance. Generally, a maintenance and evolution phase follows the Acceptance and Finalization. This phase is concerned with modifying the existing system to meet new requirements. However, in our case this falls beyond the scope of the project and contract, so we conclude this project with the Acceptance and Finalization phase.

Having completed all the preceding steps of the software engineering process that have been described in this document, we now have a tested prototype and a thesis describing the underlying research on which it is based. Though we are very proud of what we have made, and our client too is very satisfied, it would be an illusion to think we can burn the program on a cd and drop it in my client's pigeon hole. Obviously it would be possible, but fully dissatisfactional for us, our client and our future users. Without a correct turnover, the product may never be touched or looked at, and shortly forgotten. Obviously that is far from our intentions, and we devised a plan to ensure that the product can and will be used and remains viable within IBM.

8.2.1 Goal of Acceptance and Finalization

The goal of acceptance and finalization is to turn over an easy to use tool that supports children in their search for information on the Internet to our client.

In order to achieve our goal, to support children, the product we have made must reach the children. This can only be achieved if the project remains viable after we leave and finish off our internship. We have placed our heart and sole in this project and would be very honored if someone could pick up where we left and finish off what we have not been able to complete, add their own new ideas to the existing prototype, or incorporate the ideas our prototype demonstrates into a full version product ready for launching. Furthermore, our client has indicated that he shares this passion with us and intends to try to keep the project and its results sustainable.

We have not forgotten that our client is the manager of a marketing division, not a computer scientist, and not willing to spend time and effort trying to get the system up and running. In order to ensure the viability of our ideas, the prototype we have made must be up and running. As such, our client will be able to demonstrate our ideas to a large public, as not everyone will be interested in reading this thesis. Increasing the potential for our client to arouse interest and awareness increases the potential for the project to be carried on.

An essential part of the finalization of a project is evaluation. As such we not only evaluate the project results (in terms of product as well as academic research), but also evaluate the process with which we achieved these results.

This section has been written with several goals in mind. The final evaluation is indispensable in an academic research, and describes on a high level what our research results are and justify that these are valid, qualitative, reproducible and integer. Furthermore, this section describes our client's expectations regarding product turnover, and acts as a checklist to ensure that we finalize our internship accordingly. Also, this section can be interesting for any person who wishes to continue elaboration on our project. It includes not only a list of all components yielded during turnover, but also indicates which parties are aware and have shown interest in our project, both a vital starting point in initiating a new project with the intention of elaborating on existing components or ideas.

8.2.2 Acceptance and Finalization Process

As a part of the acceptance and finalization process, together with the client, steps for acceptance have been devised and completed to make sure that the product is embedded and can be fully used, as well as building awareness for the product among all stakeholders involved (users, parents, teachers and IBM) to ensure sustainable results. The product results that must be turned over to make full acceptance possible are described in section 8.3. Aspects pertaining to awareness are described in section 8.4.1.

Furthermore, during this stage the thesis will be finalized and presentations will be held at the KUN and IBM describing the project and its results. A final evaluation of both the process approach (see section 8.5.2) and the tool (see section 8.5.5) will be described.

8.3 Product Turnover

To ensure that our client is given a product that will not be hidden in a closet, but shown and used, we have devised a plan to ensure this. The prototype is to be fully-installed and ready for use onto a stand alone computer. Along with this we will hand over a CD with all necessary information pertaining to this project and the requirements and instructions for installing our program onto another computer:

- Fully working system on an IBM laptop;
- A short introduction to the contents of the CD (a 'readme' file);
- All source code files and documentation;
- Printed instructions on how to start the program;
- Installation guide;
- This thesis;
- Rational design model.

8.4 Business Requirements

Section 3.4.2 states that there were two business requirements, namely raising awareness and a multi-linguistic software product. This section describes the compliancy of this project to each of the requirements.

8.4.1 Raising Awareness

The Requirements Analysis states that one of the business requirements is raising awareness within IBM, at schools, and in the community (see section 3.5.1). During the functional review (4.7.1) we realized that the business requirement pertaining to building awareness is a separate process that is not to be accomplished by the tool, but by the project owners. Together with the client we have devised a plan to increase awareness of our project across the globe. The following initiatives have been taken:

- **IBM:**
  - Wrote an article for IBM's w3 intranet to build awareness under IBM-ers;
  - Give a presentation and workshop explaining the results of what we have accomplished to build awareness for IBM-ers and our client's contacts;
  - Contact with IBM-ers across the globe (United States and England);
  - Increased awareness at IBM by making and hanging up posters communicating our initiative;
- **School:**
  - Taught a computer course at a middle-school to increase awareness of the problems and possible solutions (including this project);
  - Published an article in a middle-school paper to build awareness among students, parents and teachers;
- **Community:**
  - Registered the project with an organization called "Ict Op School" whose goal is optimize soft and hardware use of ICT at schools;

– Gave a presentation at the Radboud University of Nijmegen to build awareness under university students, teachers, among others;
– Increased awareness in the community after publication of an article mentioning the project in the newspaper 'De Gelderlander'
– Increased awareness at the university and found two students willing to continue with our project in the near future.

As described in section 4.8, because this business requirement could not be incorporated in the design, it has not followed the standard procedure as all other requirements have. It can therefor not be tested in the same manner, and an alternative manner for verification must be devised.

In order to verify that the building awareness (requirement B1 from section 3.5.1) has been completed, we refer to our client, as he is the soul person who established this requirement. Upon verification he has approved the result. We conclude this requirement has been adhered to and validated.

8.4.2 Multi-linguistic

The other business requirement was to yield a software product that was multi-linguistic, capable of dealing with English, Spanish and Dutch in aspects of document retrieval, query input and interface. During prioritization (section 3.5.1) it was indicated that only performance in English was to be a primary requirement, and Spanish and Dutch were to be secondary requirements. The primary requirement has been designed, implemented, tested, validated and verified. The secondary requirement has only been designed. Conclusively, this business requirement has been achieved.

8.5 Evaluation

A final evaluation is indispensable in any academic research, describing research results on a high level and justifying their quality. The systematic process with which we attained and documented our (research and tool) results show that our approach is valid, qualitative, reproducible and integer, as the reader can verify all steps taken. The following sections evaluate the academic research which formed the basis for this thesis (section 8.5.1), the process with which we came to our results (section 8.5.2), the limitations that played a role during the investigation (section 8.5.3), an evaluation of the quality of the results (section 8.5.4), and the tool that was a product of this process (section 8.5.5).

8.5.1 Research Evaluation

Besides the main project goal (see section 1.2.2), to develop a successful product, we have distinguished several research goals in order to make our project goal possible (see section 1.3.2).

Management and Technology Research

The goal defined for management and technology research pertained primarily to the requirements analysis. After having defined the goal, it was analyzed and a plan was set up in order to systematically answer the research question. All steps, results and conclusions have been documented in this thesis. As such, any person wishing to check our work can verify that the conclusions we come to are valid, reproducible and integer, and thus qualitative.

Specifically, the following has been analyzed:

- *Who were the stakeholders?*
  Investigated during Requirements Analysis phase and documented in appendix A.1. The results were used for determining who we needed to involve and consult in order to determine all the relevant requirements posed to ensure a successful product.

- *Which problems are encountered by each of the stakeholder groups, and to which extent and under which circumstances do they consider it to be a problem?*
  Investigated during Requirements Analysis phase and documented in appendices A.1, A.2, A.3, and A.4. The results were used to determine if there were any problems (and thus a need for a solution) at all, and if so, what problems were at hand that needed to be solved.

- *Which requirements do the different stakeholders involved place on a tool to help deal with these problems?*
  Investigated during Requirements Analysis phase and documented in appendices A.1, A.2, A.3, and A.4. The results were used to determine what the stakeholder requirements were, which was in turn used to make the specification of the product to be built. The stakeholder requirements are listed in section 3.4.2.

Computer Science Research

The goal defined for the computer science research pertained to all steps of the software engineering process and the information retrieval research; requirements analysis, design, implementation, testing, and acceptance and finalization. After having defined the goal, it was analyzed and a plan was set up in order to systematically answer the research question. All steps, results, and conclusions have been documented. As such, any person wishing to check our work can see that the conclusions we come to are valid, reproducible and integer, and thus qualitative.

Specifically, the following was to be analyzed:

- *Which relevant problems can be solved by the implementation of a tool? What are the origins of these problems and how can they be solved?*
  The problems and their origins have been examined and described in appendices A.1, A.2, A.3, and A.4. Those problems for which a tool could be a solution, formed the input for GUI research (section 4.6)and IR research (section 5.3).

- *Is it possible to adhere to their needs by increasing retrieval precision, delivering relevant documents and filtering irrelevant documents?*
  Research into retrieval information, current research and capabilities formed the answer to this question, documented in sections 5.3 and 5.4.

- *What are the specifications of an on-line multilingual application which is intuitive enough for children to work with yet strong enough to aid them in their search for information on the Internet?*

Translated from the requirements analysis a formal product specification was designed and modelled. These models are described, including justification of the translation methods, in chapter 4.

8.5.2 Process Evaluation

Throughout the project we have made decisions pertaining to approaches and tools that we would use. In this section we discuss the choices, their implications and their effects. As such, any person working on a similar project may be able to learn from our experiences.

Developmental Model

- **Description**

  In our opinion, good software project management is essential if a software engineering project is to be developed on schedule and the result is to meet client expectations. Furthermore, a systematic software engineering process is necessary in order to a yield a qualitative product.

  A developmental model forces software engineers to structure the software engineering process; starting from a complete a requirements engineering, then systematically translating each requirement in the specification into a design, and then again into code which results in an output that can be tested, verified and validated. Quality is not only determined on the characteristics or behavior of the final product. Structure in coding and modelling determine its capability to be adapted, tested and reused [10]. Therefore, more important in quality determination is the process with which the product was built. A well structured and transparent process during which all steps and decisions are documented allows for reflection and quality control. Furthermore it ensures that no steps are omitted and that the complete output of one phase forms the input of the next.

  In our Project Plan (see section 2) we established that we would follow a software engineering method called the waterfall model (see figure 2.1 for a graphic representation of the model). The waterfall model is an iterative process model with feedback loops that presents the software process as a cycle of activities. During requirements analysis we developed a software specification. In the subsequent phase, the design process was concerned with systematically transforming the requirements specification into a more detailed specification for the system's developers which was transformed to an executable software system during the implementation process. The testing process incorporated validation during which it is checked if the software system conforms to its specification and that it meets the real need of the users of the system. Though we based our process on the waterfall model, we have adapted it slightly to our liking. As such, following each phase we have added a testing and evaluation step during which we verified:

  - the output of the previous phase formed the input of the current phase;
  - the input of that phase had correctly been transformed into the output;
  - all steps made had been documented and could be verified;
  - all steps made were valid and logical.

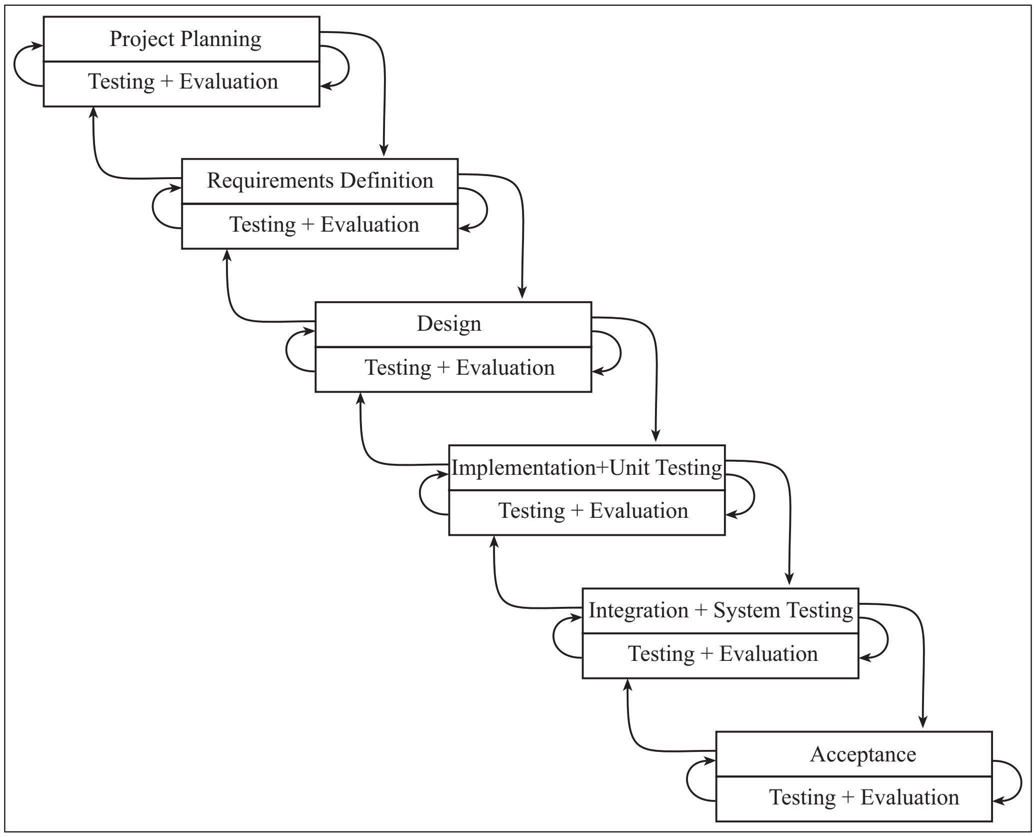


Figure 8.1: The adapted Waterfall Model

If this was not the case, a feedback loop was made back to the start of the phase to adapt and then evaluate the phase once again. In effect, the adapted waterfall model that we used has graphically been depicted in figure 8.1.

- **Advantages**
  A clear advantage of the waterfall model is that it is a simple management model, and its separation between phases strongly reduces complexity as it allows the engineers to focus on only one developmental aspect at a time. It is furthermore a very structured, systematic and transparent approach to building a software product. This structure indicates clear points for milestones and yielding deliverables, and has also structured us to write our thesis incrementally. We have scheduled milestones to occur regularly throughout our software project, in each case yielding reports which affirm and communicate project progress to our supervisors and client. These milestones easily coincide with the phases of the waterfall model, and thus creating them almost comes natural. Reports help to establish common ground and clear expectations between both project owners, client and supervisors. Considering that the problem (and the solution) was well-defined at the beginning of the project, and we sought a well-structured method to avoid losing track of time, the developmental method seemed like a rather good choice.

  The waterfall model allowed us to cascade between consecutive phases upon encountering an error in a previous phase, without having to restart the entire software process.

An example is the straightforward process with which we could adjust the criteria with which we test the successfulness of our product. The criteria that had been established during the requirements analysis phase (see section 3.4.2) had been sharped during the test phase (section 7.6) to ensure that only tested, verified and validated results could be counted as successful (rather than merely the fact that it had been implemented). Cascading between processes forces the engineers to think about the effect of a decision on other parts of the code. This showed to be useful when the requirement for an 'Advanced Search' option was dropped because we had learned that the users were unable of understanding boolean syntax. We had to take a step back, from the design phase back to the requirements phase and determine which other aspects may be effected by this new information. We had realized that the specification of the 'Query Input' was affected by this choice and had to be adjusted to ensure that it did not deal with boolean syntax (a new assumption was made that multiple terms were merely seen as 'AND' rather than allowing boolean syntax as input). After verifying that no other requirements were affected we continued on to the design phase in which we checked if our adjusted requirements altered any of the design models or choices. Note that this decision not only affects the 'Advanced Search' option available to the user, but also the manner in which queries are compared to documents. The waterfall model forced us to go back, deal with the new information and determine its impact throughout the entire system. Though it was time consuming it ensured us that the requirements did not conflict, and that the adjusted requirements were taken into account throughout the remainder of the software engineering process. A change in the requirements can have a large impact, and all related requirements and consecutive models must be checked prior to continuing.

Another advantage of the waterfall model is that one phase is completed prior to initiating the following. This indicates a clear points in which we can reflect back on the phase, forcing us to get feedback from our client, users and supervisors, and determining whether the project is still feasible, prior to continuing to the next phase. For example, upon completing the design phase we realized that implementing and testing all primary and secondary requirements would not be feasible considering the time remaining. We thus adjusted our approach to the implementation phase by deciding to focus on the primary requirements and leave the rest for later. Possibly, without such a moment in which reflection can take place, we may have not been able to make such a timely decision.

Furthermore, the output of one phase is the input of the next. The main advantage of such a process is that it can easily be verified that all the information from one stage is incorporated in the next (as we have consistently reflected in the results section of every chapter). Doing so has automatically led to a table describing which requirements have or have not implemented, including justification of those results, an important input for the evaluation of our product.

- **Disadvantages**
  As the description of the waterfall model indicates, this development model requires commitment to a set of requirements prior to design, and commitment to design before implementation. Changes in the requirements during development require rework of the requirements, and also all the successive phases such as design, implementation and

testing. This has shown to be rather frustrating, because upon determining a design flaw during testing we had to go back as far as the requirements to fix the problem, and from there incorporate the changes in the implementation and testing phases again, as well as all the accompanying documentation. An example of this was the cooperation between the directory and the relationship-map. A selection in the relationship-map was to adjust the contents of the directory. However, terms are unambiguous and can belong to more than one directory category simultaneously, hindering visualization. This error was made during the requirements phase and had not been seen until the implementation phase.

A clear disadvantage of the waterfall model is the large time-lag between making a decision and seeing the result or impact of the decision. An example is the use-case model which was made during the requirements analysis phase and had to be adapted much later during the design phase to incorporate all the new requirements. In the same line, most of the errors in the requirements or design will come to light during the test phase, resulting in an expensive trajectory for debugging. We have indicated, due to time constraints that a full test of our prototype has not yet been feasible. As an effect, we expect that the most bugs and errors will come to light at a later point in time.

However, though encapsulating phases into distinct parts has shown to be useful, it is not natural. For example, during the design phase we had learned from users that they were not capable of understanding boolean syntax, and that the problem was so big that a simple interface solution could not deal with the problem, dropping the 'Advanced Search' option. The input of information from users or other stakeholders does not constrain itself to the predefined requirements analysis phase of the waterfall model, but occurs naturally throughout the entire project.

Considering these issues, possibly another developmental model (i.e. rapid prototyping or the spiral model) could have been a better option. Though any model has its advantages and its drawbacks, a prototyping developmental method shows faster results which can be shown to the users for feedback, increasing coupling with user expectations. However, such a method has the drawback that it often yields a less robust code structure (something our client would not be happy with, or any person intending to continue on with our code), less structure for management, and possibly traceability issues.

- **Reflection**
  Looking back on the developmental process, we realize that we have adapted the waterfall model slightly. At the end of each phase we reflected the output back onto the input of the phase to ensure complete mapping, traceability and a correct translation from input to output. At each phase we thus reflected on aspects pertaining to the phase itself, including processes and methods, rather than only results. At the end of each phase, we also verified that the complete list of requirements had been incorporated in the phase's result. As such, cascading occurred not only in between phases, but also inside phases.

  Thus, though the model we choose to use had its drawbacks, we still have the opinion that it was a good fit (after a little moulding and adaption) under the circumstances and for the purposes for which it was intended.

Requirements Analysis

- **Description**
  Prior to requirements analysis we completed a literary study ([22]) about different methods of requirements distillation. In distilling requirements from the stakeholders we used several different methodologies depending on the characteristics of the stakeholder in question (i.e. adult or child), the type of question we had (i.e. specific or vague), and the type of information we sought (i.e. fact or opinion). Specifically, the methods have been described in each of the Appendices A.1, A.2, A.3, and A.4.

- **Advantages**
  The advantage of using several methods of requirements distillation is that you can choose a method that seems most appropriate for the specific source and the type of information you wish to distill.

  As such, while attaining requirements from children we combined interviewing with observation, yet also allowed them to fill in a survey. Due to technical restrictions on the hardware at the school, the screen recorder which was to record their movements during a test search had failed. However, not all efforts were lost because the survey gave us a lot of extra information about what type of problems they ran into.

  Furthermore, after having established the initial version of the requirements, we sought feedback from our stakeholders by discussing our findings. Each stakeholder was approached differently (i.e. a list of requirements or a prototype interface). This ensured us we had not lost any information along the way.

- **Disadvantages**
  During observations with the children, we realized that the groups we had chosen were actually far to large to notice and document all their behavior and responses to certain situations (i.e. if a particular feature irritated them or not). Smaller groups should have been chosen in order to take note of all reactions. This is very important for the particular source, because, as we have noticed, children are not always capable of reflecting on their own manner of working.

  Though our approach seemed systematic, the requirements we distilled were not complete. For example, during requirements analysis at IBM we had missed out all turnover requirements.

  Requirements which seemed viable initially showed to be incorrect at a later point in time. After seeking feedback from our client we dropped them. An example is the 'Advanced Search' feature which seemed to be an interesting feature initially. However, as we got to know and understand the children better, we realized that simple implementation of the feature would be useless without additional education as Boolean functionality is far too complex for a child of our age group to understand.

- **Reflection**
  Though not flawless, we are very satisfied with both the approach taken and the results yielded from the requirements distillation process. In combination with the developmental method, incomplete requirements were adjusted at a later point in time.

Design Tools

- **Description**
  We have chosen to use methods standardly used by IBM in software development, for our design. This included ORM, UML, CRC and the use of IBM's Rational Rose tool. The goal of using these formal techniques is to ensure correctness of the description, reduction of ambiguities and inconsistencies, validation of the architecture and an increased legibility and overview of the description. The result is a specification which can easily be verified and translated for implementation purposes. This forms the input for the implementation task that is to occur in the subsequent developmental phase.

- **Advantages**
  Using a structured method for design had its clear advantages. For the most part it ensured that we thought about what we were going to do prior to implementation. We were forced to think about and justify architectural decisions which would create an adequate infrastructure for our product.

  We were able to break the large and complex product we wished to build, into smaller components that could be implemented and tested individually. More importantly, using the CRC cards, a specification of each class was derived which was used for further modelling and finally as a recipe for implementation. We divided these classes between the two of us to define who was going to implement and test what, without us both writing the same functionalities and thereby doing the same work twice.

  By determining and modelling the manner in which the components interact, we could ensure that the components had a high cohesion and low coupling, simplifying the process of integration (and testing thereof). Moreover, this approach simplifies the introduction of new components that may be designed and integrated in the future, and thus increasing maintainability.

  By describing how the system was to work we were forced to think about how we were going to deal with several aspects. An example is communication with the database, another is the manner in which the incoming documents were handled and analyzed (describing sequences of functionalities, i.e. first parsing, then stripping and then stemming document contents). It also ensured that the two of us had the same idea of what we were building.

- **Disadvantages**
  Though the models we used are in essence very powerful, they are only powerful if used to their full extent, and furthermore the diagrams cost very much time to create and modify.

  We used the models for the initial designs and high-level modelling. However, after specifying the classes and the sequence diagrams of the most complex functionalities, we stopped using the models.

  During programming we realized that some errors had been made in the models. For example, the document preparation was modelled to determine if the document was black towards the end of the process. At the time this seemed natural because determining the group followed the determination of prominent concepts in a document, which could easiest be done if the document had already been transformed to a document vector. However, after stemming, determining if a document is black yields very imprecise results as the black terms can not readily be recognized as such. Thus we had to choose an alternative approach to what we had designed. Another example is the Database class which, according to the model was designed to contain only a handful of functions. However, for practical purposes, the functions themselves were either broken down into smaller functions or required other functions to complete certain calculations. In the end, instead of some 10 functions, the class has a length of 3000 lines of code and contains more than 50 functions. Specifically, updating a table in the database prerequisites that we find a corresponding entry and change its value, and only if it does not exist, we must insert the value into the table. While building the design we abstracted from such details, which was necessary to maintain overview and get the big picture. At the time of implementation this meant that such procedures had to be thought of and built. Though these errors were dealt with in the code, it is very expensive to go back and adapt all the relevant models, and at a particular stage we decided not to adapt the models anymore. The result is that the models do not reflect the current state of the product anymore and cannot be solely used to base further elaboration of the product on.

Also, some rather unpractical class distinctions had been modelled. For example, the Lexicon and the Analyzer were forced to collaborate much more than the design specified. The Analyzer was modelled to determine concepts in documents, while the Lexicon was to determine concept groups. In practice, we juggled the functions around slightly and all functionalities pertaining to language or concepts were moved to one class, the Lexicon.

As creating and modifying the models are a very time-consuming task, we have chosen not to model all of the functionalities in low-level diagrams. A reason being that we would spend time designing details of secondary requirements which we may not even implement. We had incorporated all requirements into the high-level design, assuring further more specific designing and implementation of the requirement was possible at a later point in time. Furthermore, we were not planning on using the automatic code generation tool that Rational has because from experience we know this code is inefficient and not legible to a human, leaving few advantages to completely modelling every aspect of the system on a highly detailed level. Because the model was not completed into detail, there was no reflection on the completeness of the design models, and the requirements themselves had to be used for validation of the phase.

- **Reflection**
  We have seen that the design models were very useful to analyze the system we were to build and make clear to ourselves how we would attempt to do so. It broke the problem down into far less complex sub-components, making a clear division of tasks between the two project owners possible. However, the design models are theoretical models, and show to be impractical at times when a physical implementation is being made. Furthermore, as they are very time consuming to be made and adapted, the required functionalities have not been described into such low-level descriptions that they could be translated to code on a one-to-one basis. And as such, only the high level design models could be used to verify that the code was correct (as described, the lower level models contained many errors or implementation deviated from the model for practical purposes). Instead of using the models (for which they have been designed), validating

and verifying the code had to be done using the initial requirements. Conclusively, we indicate that this modelling method can be useful in very large project groups or when dealing with very complex systems, however, for the system and project group that we were building, maybe a simpler method could be used during design which costs less time and effort to set up, or possibly one could choose for another developmental model (such as rapid prototyping) which does not require specific design models.

Version Management

- **Description**
  Both project members had to share code and documents. In order to assure both members had the most recent version of each, a version management system CVS was used.

- **Advantages**
  The clear advantage of CVS is that the newest version of each document or code product is available to both project members at any moment in time, from any location in the world. Manual version management is not required anymore, relieving the project owners of a time-consuming task. Logs are made to indicate which changes have been made and by whom. Furthermore, backups are always made of all documents, and the server on which all the documents are stored makes back-ups on a daily basis.

- **Disadvantages**
  The disadvantage of CVS is that it requires an internet connection. When working in a location without internet, there is a lag between the time updates have been made and the recent versions have been committed. Furthermore, if two people adjust the same document simultaneously, an error is reported and the document must be reviewed by hand, a time-consuming activity.

- **Reflection**
  We would advise any person or groups of people to use CVS, or a similar tool when having to share documents. It ensures that the newest version is always available to everyone and relieves the burden of version control. Furthermore, there is always a back-up to fall back on if you do something stupid (trust us, we speak out of experience!).

Database Design

- **Description**
  In order to determine what type of information was to be stored in the database, and which tables were required in the database in order to do so, we made use of the Object Role Model (ORM). An ORM diagram (see figure 4.3) was established by creating natural language sentences which describe the use-cases and system use-cases, and then translating these into the model. The components in the model are type-checked, grouped, lexicalized and reduced into a normalized relational information-model (describing tables, fields and their relations) which can directly be translated into a database design.

- **Advantages**
  The advantage of this method is that, under the assumption that correct translations

have been made, a consistent and concise database design is yielded. All tables are type-checked and constraints are checked for consistency. As the description is a list of tables, their column names and their field types, this can really be used immediately.

- **Disadvantages**
  Translation of the information of the use-cases to the natural language sentences, which formed the input of the ORM, showed to be problematic. Though the ORM is checked for consistency, it lacked systematically mapping to the original use-cases to ensure that it was valid and complete. We had discussed the resulting ORM in relation to the use-cases to consider if it was complete, however, the resulting ORM is so large and complex that we had overseen issues resulting in unforeseen complications which we now discuss.

  An example of incorrect modelling of established requirements is a table containing the directory categories required for the directory listing structure. It was mentioned in the requirements, however was not incorporated in the universe of discourse and was thus lost in the final ORM.

  Inappropriate establishment of database requirements resulted in a database that was unappropriate for our goals. Examples are that we had not foreseen that document frequencies were stored as values between 0 and 1 rather than increasingly large terms, or that query vectors would also have frequencies.

  Not all complications were a result of incorrect modelling of the established requirements or incomplete requirements analysis. During programming our database needs had changed. For example, besides the tables containing the terms used in documents and queries (which was needed to fill the relationship-map), we realized that we needed to store the stemmed variants of terms as well (document phrases were stored in their stemmed version for efficiency purposes and to aid the user by mapping term variants to the same term). Another example is the capacity of a field. We expected 8 bytes to be large enough to store identification codes for phrases. However, in practice the database rejected many codes because they were to large, so we had to change the field from `int` to `bigints`.

  Upon working with the database we had to change tables, and their fields very often. As we were working with two different databases (one database for each of us), ensuring that we were both working with the same tables was very time consuming. We even had to create a separate file to indicate which tables we were using. Even though this is also needed for the installation guide at product turn-over time, we would have preferred to deal with this only once, and not have to constantly adapt the list of tables. The effect is that we had to repeat the feedback loop between requirements, design, implementation and testing multiple times before we had established an adequate database design.

- **Reflection**
  Conclusively, we should have used a more structured and transparent method, model or tool that could aid us in analyzing the database requirements and then translating these into an appropriate database description. This would have yielded a much more correct and complete database description prior to implementation.

Programming Tools

- **Description**
  The database we chose to work with was DB2. As this is IBM's primary database we were asked to use it. DB2 functions only with SQL and Java, so the decision to use DB2 implicitly forced us to use these.

- **Advantages**
  DB2 has efficient algorithms embedded in it for searching through and storing tables. This is advantageous as it shielded us from such implementation issues.

  Furthermore, DB2 is extremely powerful. In case of expansion of the system, DB2 has a strong enough architecture to deal with that, as long as it can be run from one computer.

  As DB2 is an IBM product, there were many experts available to help us. Two DB2 experts gave us a crash course on the use of DB2 and helped us make architectural decisions pertaining to the system. Furthermore, there is a lot of documentation and troubleshooting information available about DB2.

- **Disadvantages**
  DB2 is extremely large and complex, and requires quite some time before you learn how to use it.

  As DB2 is only attainable through licensing, we could only run it on the laptops provided to us via IBM. The large disadvantage here is that you are restricted in hardware. For example, one of our laptops broke down, leaving us with only one working database until replacement had been found.

  Parallel programming cannot be attained using DB2. This means that in situations in which one wants to use multiple processors to improve performance, the architecture is incapable of dealing with the demand.

  Because we had DB2 running on each computer stand alone, we had to synchronize the databases regularly. This was a very time-consuming process.

  The choice to use DB2 forced us to use Java. Java is rather slow at start-up and can show to be problematic when the product is to be launched onto the market.

- **Reflection**
  Generally any database could have been used (SQL, MySQL, Oracle). We were unable to test the speed of DB2, so we have no way of determining if it is timely enough to handle running an online search engine. Though DB2 was unmanageable at times, it was capable of delivering the functionality we expected from it.

Testing Code

- **Description**
  We distinguish between two types of testing, phase and code. Phase testing considered aspects pertaining to the input and output of each phase and especially the transformation in between. For a discussion on this topic we refer to the section about Developmental Models in this chapter.

The testing procedure of the code was interleaved with implementation. Unit testing followed implementation of each function and class. Afterward, integration and system testing took place. Conclusively validation, verification and evaluation of the entire system took place. Initial testing of the interface occurred during the design phase.

- **Advantages**
  The main advantage of unit testing is that functions and classes are tested immediately. Pinpointing bugs is an easier task because it is restricted to the code being tested, rather than testing the entire system and looking for a needle in a haystack. Furthermore, interleaving implementation and testing is advantageous as programmers can immediately learn from their mistakes. An example is a syntax error in an SQL code (as in our case, mistaking ‘ for ’) which is immediately noted, rather than having to adjust all incorrect SQL codes afterwards.

  Testing the GUI prototype at a very early stage in the development process (the design phase) was very useful because we got feedback from our users and were able to incorporate that into our final mode, ensuring that the interface adheres to the users needs and way of working. It also tests that all required functionalities have been checked.

- **Disadvantages**
  The disadvantage to this method is that verification and validation can only be completed at the end of the phase. The time-lag is large, and much work has to be redone if an error is encountered. Furthermore, fixing a bug near the end of the phase requires retesting of all previously tested items, as due to integration, altering one function can result in errors in a completely different class. Integration testing is also very complex in Object-Oriented programming, due to features such as inheritance and polymorphism. Furthermore, not all class functions can easily be tested prior to integration, because they require integration to become fully functional.

  Testing lacked procedures such as testing on improper input, capacity testing, performance testing, responsiveness testing, or what would happen to the system under extreme circumstances. Furthermore, α and β testing would have been very useful to determine how happy the users really are with the system and encounter usability-related bugs.

  Though logging of errors and test results should be noted after completing tests at any level, it has sometimes been forgotten. The result is that testing a particular unit sometimes occured more than once. A more structured and complete manner for approaching the test phase is desired.

- **Reflection**
  Previous to the testing phase we had completed a literary research in which it became evident that there are very few structured manners for testing OO programs, and given the problem at hand this method seemed to be most adequate, especially due to its simplicity, transparency and traceability. However, I would advise any person in the same situation to check if there *really* is no other method more appropriate. We did not devise specific tests during the design phase, which would probably have given the test phase more structure. Furthermore, a better and simpler manner for logging errors is advised, possibly by using some sort of tool rather than tables in LaTeX which are actually not suitable for this purpose.

8.5.3 Research Limitations

Obviously, the most restricting constraint we had to deal with was time. Because we only had a half a year to get everything done we had to place strict time constraints on each of the software engineering phases. Also a limitation was (wo)man power, as the project team was composed of only two people. Another limitation was hardware. As the database could only run on the IBM-licensed laptops, we were restricted to implementation and testing on these computers. As one of the project owners had a great fall, this resulted in loss of a laptop for a considerable time-period until it could be replaced, which cost a lot of time and effort. These were overall limitations that pertained over the entire project.

In spite of these limitations we still believe that our results and conclusions are of a high quality. Where we were not able to complete studies or spend enough resources to attain very accurate results (for example, extensive testing), we honestly documented our results, described the limitations that played a role, and evaluate the resulting quality of the study.

We now discuss the limitations of the research independently per phase.

Limitations on Requirements Investigation

We were limited in the amount of time we could actually spend with children to learn what goes on in their heads, and time we could spend with IBM employees to learn how what their approach was to the analysis, design and implementation of a product, what problems they run or have run into and what they do to solve such problems. The only time that is enough in such a research is ’forever’. As this is unattainable we objectively evaluate, and, as we taught children of our focus group over a course of almost half a year, and have had many meetings with programmers and other employees at IBM, we feel the results achieved are qualitative.

Limitations on Design

Design tools are generally very expensive. As IBM has its own design tool, Rational Rose, we were asked to use their tool. This had both its advantages and disadvantages (see 8.5.2). IBM has many experts on Rational working in the company. We were able to show our models to several software programmers that had experience with the tool and get valuable feedback on them. These programmers also gave us advice on architecture decisions, which type of an environment we could best choose to work with, considering the tool we wished to build and its (future) expectations.

Though we both had experience with the models used in this software engineering phase, creating the design models showed to be time consuming. After determining that a requirement had not been designed, the design models had to be adapted which costs very much time. Especially ensuring that requirements were not only incorporated into the design, but incorporated correctly (as our future user will expect to use it) was difficult as not all requirements can be translated one-to-one into design without either affecting other requirements or the design as a whole.

Validation that all requirements were dealt with was tested, yet exact tracing of all requirements were difficult. As an example, we tested that the interface was ’easy to use’ (one of the requirements) by building an interface prototype that we showed a few children. This gave us an indication that we were on the right track, however, obviously to be completely sure we would have to have a complete functioning software program and test that on any child who may be a future user of our system. In other words, we had to narrow down testing and verification to just a handful of potential users. We however did attempt to get a wide range of users from several backgrounds to get rather objective and clear results.

Limitations on IR Investigation

Due to time limitations, it was not possible to implement every aspect which attained positive results in the final Giggle prototype. In many cases, more elaborate testing is required to ensure that the results are indeed positive. Without adequate testing, we did not wish to implement features if we were unsure about their performance. However, we have established some promising preliminary results to elaborate on for anyone wishing to continue with our project at a later point in time.

A hardware and software constraint also played a role in the information retrieval investigation. Many search engines implement parallel programming on multiple processors as a manner to achieve results on a more timely basis. We could not implement this because of the database choice we had and the two laptops we were given. It would have been interesting to implement such an algorithm to test the results, and in turn investigate the advantages and disadvantages of such parallel programming.

Limitations on Implementation Process

We were limited in the amount of time we could spend analyzing which methods to use for implementation, which existing tools we could use, which algorithms to use, and obviously on aspects such as testing and verifying that the code that was written was correct.

Limitations on Testing

As often is the case in a software engineering project, we too were dealing with an extreme time constraint. With the project's deadline in sight we had to complete testing on a more superficial level than we would have liked to. Furthermore, we were greatly restricted in (wo)man power. With just the two of us, completing both verification of all units and the system as a whole, and also validation was extremely much work. As a result, this restricted us in the amount of time we could actually spend testing and fixing errors, and as a result we were not able to fix every error we encountered.

Though we were unable to test every single statement of code after having integrated the entire system, we made decisions as how to test the program thoroughly enough to gain confidence that the software system is good enough for its intended use. As every function in the code had been tested separately during building, and we were able to test the behavior of combinations of functions in classes, components and the entire system, in our opinion we were able to gather a rather precise and complete view of aspects pertaining to testing of the prototype. At least, for the amount of time and manpower that is available.

Furthermore, in order to verify the conclusions that we have made, we have let several users and our client try out our system and comment on it. As such, in this final version, the comments of our users and client have been incorporated in this section. Conclusively, these people have agreed, as far as they are capable of determining such, that the prototype of the system adheres to their expectations.

Phase	Section
Requirements Analysis	3.5.3
Design	4.8.1
Implementation	6.5.1
IR	5.6.1
Testing	7.6.1

Table 8.1: Evaluation reports per phase

Limitations on Acceptance and Finalization

Close to the end of a project, time is always pushing, and the clock seems to tick faster than ever before. We had previously not really regarded which specific aspects had to be dealt with during product turnover. We chose not to choose for the ”over the fence” method with which the product is just given to the client. To ensure its viability we had to provide the product in such a state that it could be usable in the future with minimal efforts of the client, or else it would simply not be used.

8.5.4 Quality Evaluation

During each phase in the software engineering process, we let several people read and review the preliminary and (almost) final versions of our reports, models and prototype and comment on it. As such, in this final version, comments of our client contact at IBM, software programmers, parents, middle school teachers, children, and our supervisors at the university have been incorporated. Conclusively, these people have agreed, as far as they are capable of determining such, that the aspects described in this document are valid and complete.

Though we were obviously unable to research, investigate, and get feedback from every relevant person on the globe, we had to make decisions as to who we would investigate during each individual phase. In determining this we considered aspects pertaining to the quality of information, such as validity, reproducibility and precision. Each investigation indicates which characteristics the interviewees must adhere to, which sources we interviewed, and how we distilled information from them. In our opinion we were able to gather a rather precise and complete view of aspects pertaining to the problems in question during each phase. At least, for the amount of time and manpower that is available.

Each phase was concluded with a short evaluation indicating the quality of the results of the conclusions of that phase. Table 8.1 indicates where the evaluations for each phase can be found.

8.5.5 Tool Evaluation

Tables 8.2, 8.3 and 8.4 describe which functional, non-functional and business requirements were actually implemented in the final tool. In accordance to the requirements, the tool is not yet complete. Table 8.5 shows which requirements were dropped during prioritization during the requirements analysis phase, yet remain part of the requirements and should be implemented to attain a product which correctly adheres to *all* the requirements we were able to distill.

Nr.	Description	Feature	P/S	Result
F1	Instructions/Help	On-line Tutorial	P	✓
		Context Sensitive Help	P	✓
F2	Metaphor	Character	P	✓
F3	Initiate Searching Process	Directory	P	✓
		Example Search	P	X
F4	Formulation Information Need	Keyword query input	P	✓
		Directory	P	✓
		Relationship-map	P	✓
F5	Aid Query Reformulation	Query expansion (synonyms/spelling)	P	✓
		Specification category/rel-map	P	✓
F6	Following Progress Search Process	Modified query	P	✓
		Path chosen (clickable)	P	✓
F7	Relevance Feedback	Store intermediate results	P	✓
		Relationship query+result	P	✓
F8	Reversal of Actions	Undo/Step back	P	✓
		Redo/Step forward	P	✓
F9	Navigation and Search Strategies	Keyword query input	P	✓
		Directory	P	✓
		Relationship map	P	✓
		Corresponding components	P	✓
		Advanced search	S	X
		Query input tips	P	✓

Table 8.2: Summary of tool accomplishments in terms of **P**(rimary) and **S**(econdary) functional requirements.

Furthermore, during testing some issues were found that could not be resolved at the time. Table 8.6 summarizes the errors encountered, the features that do not work as expected and thus are not conform the specification.

Section 7.6 establishes that on the basis of the test results on each of the requirements, our tool as a whole is successful.

8.6 Recommendations for Future Elaboration

This section describes which aspects of the system can be elaborated on if the project is to be continued at another point in time.

8.6.1 Features for Future Implementation

Some of the requirements were dropped during the prioritization in the Requirements Analysis phase. This was mainly because we merely had to narrow down and focus our intentions due to a lack of (wo)man power and time. However, these users and our client would still very much be interested in seeing them incorporated in the final product.

Nr.	Description	Feature	P/S	Result
NF1	Usability	Easy to learn/use	P	-
NF2	Presentation	Interface guidelines	P	√
		Make Giggle start page	P	√
		Login and personalization	P	√
		Block popups, banners, ads	P	±
NF3	Relevance and Filtering	Filter inappropriate content	P	√
		Filter on reading level	P	√
		Filtering on other aspects	S	X
		Promote favorable concepts	P	√
		System learning from user input	S	√
NF4	Safety Warning	Educate user about safety	P	√
		Encryption user information	P	√
NF5	Hard and Software	Windows 2000 and up OS	P	√
		IE and Mozilla browsers	P	√
		Existing filters	P	√
NF6	Reaction Time	Adequate performance	P	√

Table 8.3: Summary of tool accomplishments in terms of **P**(rimary) and **S**(econdary) non-functional requirements.

Nr.	Description	Feature	P/S	Result
B1	Awareness	IBM	P	√
		School	P	√
		Community	S	√
B2	Multi-linguistic	English	P	√
		Dutch	S	X
		Spanish	S	X

Table 8.4: Summary of tool accomplishments in terms of **P**(rimary) and **S**(econdary) business requirements.

Nr.	Description	Feature
Functional Requirements		
F4	Formulation Information Need	Natural-language query input
F9	Advanced Search	Additional input fields and help/education
F10	Sitemap	Map
Non-Functional Requirements		
NF1	Usability	Multi-user
NF3	Filtering	Blocking banners and ads
NF6	Reaction Time	High performance

Table 8.5: Summary of requirements dropped.

Furthermore, not all primary and secondary requirements have been implemented. However, all primary and secondary requirements have been incorporated in the design. As such, the design and code have been constructed in such a manner that extending the current functionality with the remaining requirements is possible without having to adapt the architecture of the system. Rather, this can be achieved by merely extending the system with new classes which incorporate the new functionality, and integrating these into the system (and obviously adequately testing).

Table 8.5 summarizes which requirements were dropped, yet are still viable for implementation. For more details pertaining to the dropped requirements we refer to section 3.5.1.

8.6.2 Features for Debugging

As our project was under extreme time an man power limitations, we were not able to remove all bugs and errors from the program. We did however test and report any errors found including a prioritization according to their effect on the program as a whole. Errors with high priority were fixed immediately. Other errors may not have been fixed at the time but do not have dramatic effects on the over result. However, we recommend, prior to launching the tool, that these errors are to be fixed. Table 8.6 summarizes the errors.

8.6.3 Features for Further Testing

We have explained that we have not adequately completed testing. However, we believe that many features we have designed may very well lead to an improved product, and therefor should be reviewed during future elaborations. The following list describes which features should more elaborately be tested:

- Document Concepts
- Directory Description
- Stemming Document and Query Contents
- Self-learning Relevance Score
- Determining Black Listing

Error	**Action**	**Priority**	**State**
Directory			
Doc-dir determination not suff.	Extend training	High	X
Language Determiner			
No distinc. Spanish/Italian	Expand word lists/gram. rules	Low	X
No distinc. Dutch/German	Expand word lists/gram. rules	Low	X
Reading-Level Determiner			
Level too high	Normalize reading level	Medium	X
Relationship-Map			
Too many subterms	Determine sub terms	High	X
Resulting Document			
No warning when leave site	Add danger warning	Low	X
Spell Check			
Not capable of Dutch	Implement Dutch capabilities	Low	X
Not capable of Spanish	Implement Spanish capabilities	Low	X
Spider			
Date retrieved wrong	Fix date retrieved	Medium	X
Tagger			
Only HTML codes	Expand for multiple formats	Low	X
Logout not possible	Logout functionality	Low	X

Table 8.6: Errors recommended to fix.

- Performance and Responsiveness issues

8.7 Conclusion

The goal of this section was to turn over to the client, an easy to use tool that supports children in their search for information on the Internet.

The previous chapters focussed on deriving the required information in order to build the tool, as also followed the steps of a process in order to actually build a prototype of the tool. Section 8.5.1 evaluates the underlying academic research which has lead to the final result. Section 8.5.2 evaluates the process with which the tool was built, from beginning to end. And section 8.5.5 evaluates the resulting tool itself. As such, these chapters together describe and justify the quality of the final result, which is a collocation of the steps and procedures that have been followed in order to get to that result. Furthermore, this chapter reflected on the primary business requirements posed by our client (section 8.4.1), and ensured that the product turned over complied to certain requirements in order to ensure its viability (section 8.3). Pertaining to the last aspect, we realized that we there were more requirements posed on the product than we had expected. However, this did not pose any problems.

Chapter 9

Discussion

The aim of this project was to successfully design and implement an easy to use tool which supports children in their search for information on the Internet.

The above aim has been described in section 1.2.2, during the initial phase of this project. Concluding the thesis, this section discusses the results of Giggle with respect to this aim.

9.1 Method

We have designed and implemented a prototype of the tool in order to evaluate if our goal has been achieved.

The first step in order to successfully design and implement an easy to use tool which supports children in their search for information on the Internet, is a requirements analysis. Determining which requirements are posed on the tool by all stakeholders involved in order to assure that the tool to be built adheres to the wishes and needs of the stakeholders, those that will use it or are otherwise involved. Others (merely than only the users) may affect the way in which, and if, the product is to be used. Some examples of stakeholders and their role of involvement are: the client whose reputation may be influenced by failure or errors in the product, a parent who gives permission to a child use it, or a teacher who encourages that it is used by their students and thereby increases product awareness. Other requirements are guidelines and procedures that are to be followed to ensure a qualitative product, imperative if it to be successful. With the cooperation of the stakeholders, relevant requirements were prioritized, resulting in a specification and design guidelines.

The specification was translated into a formal proven design model. Architectural decisions were made based on the design guidelines and our experience with respect to system design. The design model was evaluated and validated to ensure it had all the features and characteristics posed in the requirements. A prototype interface was devised, and together with users it was evaluated to ensure that the system had the functionalities that were expected from it, and that it was indeed easy to use and intuitive to work with. Comments were logged, the interface adapted, and reflected back to the stakeholders for final approval.

The design specification was then systematically translated into code. Code units were tested and integrated into clusters. Clusters were tested and integrated into the final system. The system was tested, verified, validated and evaluated. This was done by us, the client, 5 users and 2 parents. The reactions were positive.

We had established requirements with which we could test if our product had indeed been designed and implemented successfully. We verified if the requirements were available in the tested, validated and verified prototype. Our prototype passed the minimal requirements needed to call it 'successful', thus we have succeeded in our aim to successfully design and implement an easy to use tool which supports children in their search for information on the Internet.

A package was prepared for turnover to the client, in a form useful to him. This included a working prototype on a stand-alone computer and a CD including documentation and this thesis, information necessary for installation or adaption of the current program. It also included an awareness campaign to spread word about the initiatives taken and the results of the product.

9.2 Scientific Relevance

For academic purposes, the goal of this thesis is to show that we are capable of dealing with new problems on an academic level, on all aspects from approach to results. As such, section 1.3.2 describes the scientific relevance of our project in terms of the two focusses of the master diploma we hope to be awarded, namely Computer Science and Management and Technology. In the following sections we will recap and justify how the aims proposed have been achieved.

9.2.1 Management and Technology

Our extensive marketing research has led us to understand which parties are involved when it comes to children and internet, but more specifically, which problems which parties encounter, to which extent these play a role and under which conditions they become evident. With these issues in mind we contemplated whether a programming tool could alleviate any of these problems, to which extent this could be done, and which restrictions or requirements would be posed on such a tool.

Our results show that such a tool is desired and could indeed be useful. Talking, reading, writing, watching, and interviewing stakeholders and scientific work had led to a plethora of requirements and restrictions. With the help of the stakeholders we made compromises between conflicting requirements and prioritized these in order to make way for a feasible and realizable plan in order to achieve our goals.

As a result we summarize that we have indeed developed an idea for a tool which could help alleviate some of the problems at hand. It would be naive and wrong to think that all problems were solved. Though our solution attacks several aspects there is yet much work to be done. Our prototype forms a basis for further investigation. By offering a usable system we can test and reflect to see which problems have actually (rather than theoretically) been solved, which are yet to be dealt with, and if any, which new problems have arisen due to the introduction of our tool.

9.2.2 Computer Science

With the results of the Requirements Analysis in our hands and a theoretical plan with which we planned to make the cyberworld a better and safer place, we still had a long way to go. Starting at the drawing board we discussed the origins of all the problems that had been reported and which aspects could be solved by means of a computer tool. A design was made imbedding each requirement posed, leading us in the direction of a solution.

Though our education gave us a strong basis in the field of information retrieval, human-machine interaction, and software engineering, we had to deepen our knowledge by talking to and reading work of scientists in these fields. Investigations in the field of information retrieval led us to understand and appreciate ideas such as recall and precision of retrieval, and formulate and test ideas and hypotheses in which the retrieved results would more adhere to the information need of a child, and thereby inadvertently filter out irrelevant documents. Investigations into best practice methods in the field of human-machine interaction, user interfaces and visualization methods, including a cognitive analysis of our user group, has made it possible for us to create an interface that is intuitive to use, yet so powerful that it allows our users to make full use of the features we implemented. By means of best practice methods of software engineering we assured ourselves that we would not only deliver a verified qualitative product, yet be able to validate that the right product has been built for the problem at hand.

As a result, we have devised a working prototype of a tool which is has implemented aspects of information retrieval in combination with best practice methods of human-machine interaction, capable of yielding higher precision results, results which more adequately reflect the information need of our user (group). Furthermore, we have built the system to do so automatically, after initial training without user intervention (besides standard maintenance required by each software system) and self-learning which allows it to extract meaningful information from its users' actions and use that information to increase the precision of all future retrieved results[1].

9.3 Social Relevance

Knowledge, expertise and experience are in our opinions, the three most prominent characteristics of the Radboud University in Nijmegen which we represent, and where we wish to graduate from. However, our university prides itself in the manner in which this knowledge is shared and science finds its way to help society. We strive to hold this pride up high. The question that has led us to and through this research has its basis not only in a scientific field, yet has led us to create a prototype which we can use to explicitly demonstrate the social relevance and usefulness of the research we have conducted.

9.4 Investigation Results

The goal of this thesis was to describe the problems at hand when children are searching the internet, and attempt to find a solution to them. Our thorough marketing investigation has not only led to a list of problems described by many stakeholders involved (children, parents, teachers, governmental organizations), but led us to understand how children work and think. We have learned that children will not use a tool if they do not like it, and spent time learning what characteristics a tool must have in order to overcome such issues. We translated the needs and wishes of all those involved (children, teachers, parents, librarians, governmental institutes, IBM) into requirements. These requirements formed a basis for our information

[1]This works under the assumption that our user group as a whole generally seeks the same types of information and documents, and thus has the same overall impression on what is relevant or not

retrieval investigation. We devised ideas for simplifying the search process of information on the internet (specifically for children) and the retrieval of more relevant documents. In order to demonstrate the power, social value, and applicability of our ideas we implemented a prototype. These ideas in combination with the requirements were translated into formal design models, which in turn were translated into code. The final system was tested, verified and validated, not only for bugs but also its process and adherence to the initial requirements posed.

The result:

- Requirements distillation, design, coding and testing of a prototype using IBM's most valued practices and proven formal methods;
- In depth analysis of children's characteristics, preferences and cognitive behavior and translation thereof into requirements posing restrictions on the interface;
- In depth analysis of the English language to determine:
  - Characteristics depicting relevant documents for children;
  - Relationships between terms and concepts;
  - Coupling between domain knowledge and concepts;
- Improvement of visualization methods of the cyberworld, and implementation thereof;
- Implementation of information retrieval algorithms;
- Implementation of a mechanism to enforce the use of user relevance feedback without posing a heavy burden on the user;
- Improvement of the retrieval of relevant of documents due to implementation of:
  - Automatic language determination;
  - Automatic determination of reading level;
  - Noise removal by stripping common terms;
  - Automatic determination of inappropriate material (i.e. pornography, violence);
  - Automatic classification of documents into directory categories;
  - Automatic building of relationship map showing term/concept relations;
  - Implementation of user relevance feedback.

9.5 Reflections From Project Team

This section gives an indication of the division of tasks and the roles played in the Giggle project by both project owners. It must be said that in our opinion the contributions of both project owners were comparable and many tasks were executed together; we often met together to discuss ideas, checked each other's work (ideas, process, progress and documentation) on content and quality and worked side by side as full-worthy partners. Because we were both responsible for, and contributed to, every aspect of the project, distinction in our initiatives (per project owner) is only possible on a global level. However, for some aspects of the project, the initiative, emphasis and control lay primarily in the hands of one of us.

The following self-reflections have been written by each project owner individually and describe their input and role in the project.

9.5.1 Reflection Renske

In my opinion, writing a master's thesis is more than merely demonstrating my academic capabilities and justifying that I am capable finding new solutions to problems I have never seen before, and also capable of using scientific methods and techniques. The process with which I have completed working on this thesis and everything I have learned along the way has been a valuable contribution to my life-experiences.

Contributions

As described in this thesis, this project was done by two people. In order to indicate that my contribution to the project was substantial, I will describe my roles and tasks throughout the different project phases:

- *Project Initialization Phase:* Established contact with client, generated project idea, setup and wrote project proposal and project plan. Maintain communication with all external contacts.
- *Requirements Analysis Phase:* Contacted schools for investigation of children and teachers, taught children computer course, setup experiments and interviews, established list of requirements, and documented process and results in thesis;
- *Design Phase:* Designed and implemented functional gui prototype, translated requirements into CRC and UML Rational design models, investigated IR methods, and documented process and results in thesis;
- *Implementation and Test Phases*: Researched, implemented, and tested all IR algorithms and language-related functionalities. Documented process, results and decisions made in thesis;
- *Acceptance and Finalization:* Evaluated the resulting tool and research done, and process with which these results were attained, and documented this. Completed initiatives for increasing awareness. Finalize thesis and product turnover.

Experience

As I have mentioned, this project has been a learning experience. According to the subfaculty, the goal of the thesis is to show that one is capable of dealing with problems in an academic level. Also, students choosing the MT variant are to complete a traineeship in an external company in order to gain hands-on experience in a corporate environment. The best way for me to reflect my experiences throughout this project is by standing still at the things I have learned. Though these experiences (just like terms) cannot be exactly categorized into disjunct groups, for overview I will attempt to categorize them into the three following categories: academic, corporate, and personal.

Academic:

- Setting up a thesis, and learning (the necessity) to justify and document choices on an academic level;

- Working systematically;
- Dividing large complex problems into smaller ones that can be dealt with individually, and then moulding them together, and testing if as a whole, the result accomplishes the goal as expected;
- Researching, analyzing, adjusting, and implementing information retrieval algorithms;
- Completing all steps in a scientific method (e.g. software engineering process), and evaluating the method;
- Using LaTeX to write a visually appealing thesis that includes mathematical formulae.

Corporate:

- Contacting a company for a traineeship and persuading my manager of my intentions and the capacity of my idea;
- Project management, planning, and time-boxing;
- The importance of involving (interviewing and working together with) all stakeholders in all aspects and phases of a project;
- Making my expertise useful to society and understanding social implications that play a role in decision making;

Personal:

- Two-way communication with people of many backgrounds, whether a computer scientist, a marketer, a child or the guy sitting next to me in the metro; being able to listen to what drives them, and also explain what I am doing;
- Being able to feel comfortable in both an academic and a corporate environment;
- Valuing the ideas and opinions of other people;
- Working together in a project team;
- Making priorities.

Conclusion

In my opinion, I have been able to show that my academic education has enabled me to pick up new tools and algorithms quickly, find solutions to new problems adequately and document my choices and assumptions in a manner that can be understood by other academics. I hope this thesis justifies my opinion to the reader. This has been a very valuable experience and I am proud of what I have accomplished and feel that it reflects what I have learned in the last 26 years (obviously with emphasis on the last few years), both on an academic and a social level.

This project, including composition of this thesis, has made me feel confident that I have collected all the ”luggage” I need to make my way into the scientific world, whether in an academic or a corporate environment. Capable of working both independently and as a member of a group, confident in making timely decisions, keeping overview over a large and complex project and establishing and sticking to a feasible planning are some of my personal traits I am now well aware of and am sure will be very helpful to me in the near future.

9.5.2 Reflection Jolanda

One of the goals of writing this thesis is to show that we are capable to deal with new problems, which we can solve using our learned academic skills. For me, this was a very educational project. I have learned very much in all kinds of domains.

Responsibilities

In this project I have done the following tasks:

- In the requirements phase: a little research to the different methods for collecting the requirements, research to existing search engines (to find possible requirements), kept observations and interviews, and documented the results.
- In the design phase: I brought Giggle to live, translated the requirements into database tables, created a spider which crawl over the Internet to collect documents, designed the final GUI, and documented the results.
- In the implementation: translated the design of the spider, the relationship-map, the directory structure, the login, the preferences, and the result page of the search engine in to Java code, and documented the results.
- In the testing phase: tested the separate parts of the software which I have implemented, tested the overall program, and I have done some optimalizations to speed up the search process, and documented the results.
- In the finalization phase: wrote the instruction and installation guide, and turned over the product.
- Finally, during the whole project I was responsible for the technical aspects of the program, such as the connection between the database and the user interface.

Experiences

This project was very valuable for my development in this field. A lot of things turned out to go very well, but there are things that went wrong. And I am glad about that, because from failures we can learn the most. And so I come to the following list of experiences:

- Because it was a very big project, and lot of things could be done at the same time, I have learned to work systematically. One is forced to finishing first the current activity, before going on to the next one (else, one will loose the overview).
- And because this project was done in a team, I was forced to frequently communicate with my teammate, learned to share the activities (and not to do everything by my own, this project was also to big for that), and (how) to document very well the things that were done.
- I learned to work around problems, which are hard to solve, and go for solutions which not meet the requirements completely but approximates it very good. For example, I spend a lot time search for a solution for the communication between a standard HTML page and the Java applet. Very soon I found an editor field in Java which can represent

HTML pages but knows only the most common HTML tags. Actually, this was a pretty good solution, because most pages we use, and can use, in our search engine contains only the common HTML tags. But it was not the perfect solution, so a look further. In the end, the editor field has been implemented, all other trials turned down.

Conclusion, it was better to go for a suboptimal solution and go on with the project. This will save time (and time is mostly the limited factor). At the end we can always search for better solutions.

- Because of the extensive market research we have done, I experienced how diverse stakeholder think about (and handle in) this subject and how valuable that was for this project. Observations of children with different ages and backgrounds, for example, underlined the importance of involving them in this project in an early state. The brought up very different requirements, and dropped some requirements (such as natural language search queries) we found in older researches to the behavior of children with search engines.

- And just like Santa Clause, it is important to make a (requirements) list and check it twice. Don't believe everything someone wrote directly, check it for example by finding another reference.

Conclusion

I am very proud of what I have achieved. This project help to give me more confidence in what I can do. With my gained academic skills, I believe, I was able to underpin and write down my findings in an academic way, find suitable solutions to new problems, and learn and execute new things quick.

So, with the knowledge build in my study of four and a half year, I think, I am ready for the big world out there.

9.6 Summary

In short, our investigation has lead us to the following conclusion. In order to support a child in their search on internet, they not only seek a search engine that will find information appropriate to their needs and interests, but they need a tool they feel comfortable with and attracted to, a tool that has been specifically built for children. As the internet brings to the world a new dimension, children seek a place to call home in this new world. A place they feel comfortable, safe and at ease, a place they start the day off and a place to come back home to, a place where they have their own room that they can design organize as they wish in a house having all the facilities they need. They need a website like Giggle that they can mark and experience as their home in the cyberworld. The overall challenge that the user has is to specify a good query, the second is obtaining a manageable and relevant answer.

Innovative characteristics incorporated in Giggle, matching the challenge described above and making it stand out, clearly distinguishing it from other children's search engines are:

- "New home" for children in the cyberworld environment, including personalization;

- Retrieval improvement methods such as user relevance feedback;

- Multiple search-navigation methods (i.e. category listing and relationship-map);
- Aid in query formulation (spell check, finding additional keywords);
- Allow saving/storing of search states;
- System that learns what children like from the things they choose and do on the site;
- Filtering of many unappropriate materials and pop-ups.

Hope to have you all Giggling soon!

Appendix A

Requirements Analysis

A.1 Identification of Stakeholders

Prior to determining what the requirements are, the question "who determines what is a requirement" has to be answered first in order to decided where to go in order to find the requirements. This section describes who the stakeholders are that are involved in this project, and how we determined that.

In our case, the stakeholders determine what the requirements are, because a stakeholder by definition is a person or company who has an interest or share in an undertaking, in some manner involved in the process or final product. To make a successful product, it is mandatory to inquire requirements with stakeholders and to work closely with potential users of the system [21]. Table A.1 gives an indication of the location where internet users of our age group most often access Internet, indicating where we must look to find possible stakeholders:

As one can see, the most important places to access Internet are at home and at school. Of minor importance, yet still requiring attention are public libraries and someone else's home. So, the following stakeholders can be depicted:

- users (children)
- client
- carers and educators

Internet Location	**Ages 8-10**	**Ages 11-14**
own home	74%	79.2%
school	63.4%	70.8%
public library	12.2%	17.3%
community center	0.8%	1.3%
someone else's home	11.4%	16.6%
other location	0.9%	1.2%

(SOURCE: U.S. Census Bureau, Current Population Survey, September 2001.)[32]

Table A.1: Locations where internet is used.

- providers of internet locations
- socially involved

The following sections describe the involvement, role and influence of each stakeholder throughout the development or use of the product.

A.1.1 Users

The user group is comprised of:

- children of ages 8 to 12;
- normal children (typical development);
- boys and girls;
- multi-cultural backgrounds (target is Dutch, English, American, Spanish, and Latin American);
- middle class and with average computer literacy (i.e. have access to and make use of a computer, see section A.4.6).

These children have a low information literacy (i.e. they are not adept at seeking or using information), and find it a nuisance that finding relevant information to fulfill an information need is so difficult, time-consuming and aggravating [117]. We have chosen this age group because these children are exposed to the tasks of finding information on internet for school projects as well as for leisure. Younger children are often accompanied by an adult during computer use, and furthermore do not explicitly search for information or documents on the Internet making our efforts superfluous for that age group. The type of information need, language fluency and the capability of formulating a query by older children such as teenagers is similar to those of adults, and have furthermore shown to be capable of using traditional word-based search engines in the same manner as adults do. The accent for improving tools for this age group clearly lies elsewhere. Piaget has differentiated 4 developmental stages in children, specifying average age and cognitive characteristics. A short inventory has shown that our target group (ages 8 to 12), that of Piaget's third developmental stage, has considerable problems surfing the internet, as well as their carers who are not capable of continuously and uninterruptedly supervising their tasks, but are concerned about their child's safety and welfare. Children of this age group run into problems because they are merely not adept at seeking or using information, yet are quite impressionable and eager to discover reality [67]. Furthermore problematic are literacy levels, not being capable of adequately specifying a query or retrieving documents far above their reading capability, making the use of classic search engines very complicated for this specific user group. This group is also generally not able to plan and carry out a specific search strategy, but are capable of following instructions and learning new things without much repetition. During the transition from elementary to middle-school there is a shift in curriculum emphasis towards independent learning and investigation, and with Internet use becoming more prominent, feeling comfortable surfing the internet is a valuable asset in the education of children of this age, preparing them for further education.

A.1.2 Client

As project sponsor the client is responsible for final acceptance of the product. It is therefor obvious that the client's requirements must be properly distilled and that the product reflects the quality, philosophy and reputation of the client. The client should be involved at every stage of the product's development, and by giving approval of the plans and decisions made thus far, indicating awareness and acceptance of the characteristics of the final product.

A.1.3 Carers and Educators

Carers and educators (such as parents, guardians, teachers, and schools), who are involved with the upbringing of children, trying to teach them right from wrong and protecting them from inappropriate practices is a potential project stakeholder [53] [51] [54].

A.1.4 Providers of Internet Locations

Providers of internet locations are those involved with providing locations where children can use internet (such as librarians, family restaurants [52] or internet cafes) [32] and who feel a social responsibility to safety, either directly (ethical issues) or indirectly (such as image or name). Issues involving content filtering have lead to great controversy, juggling safety with unconstitutional and unethical restrictions to information which can be seen as impairment on freedom of speech. While the US government favors legislation as an approach to content regulation (such as the Children's Internet Protection Act of 2000 which requires public libraries that receive government grants to filter online content [71]) [43], the EU has recently voted in favor of a less restrictive approach, placing responsibility for content monitoring on content providers and Internet Service Providers [34].

A.1.5 Socially Involved

Another potential stakeholder set are those involved on a social level (such as IBM [7], SurfOpSafe [55] and CyberPatrol [44]), who see this as a problem and feel the need and responsibility to aid in finding a solution to this problem.

A.2 Investigation of IBM

In order to distill the requirements that makes our tool capable of supporting children in their search for information, and to be able to successfully design and implement these requirements, we have posed several (sub-)questions (see section 3.2.1) which need to be answered. The primary source in finding answers to these questions is IBM. In this section we describe how, and from who, we have collected information in order to do this, and what the concluding results of this investigation are. As our client, and being experienced in software production (also software for children), IBM is an important source in seeking answers to the following questions (see also table 3.1):

1. What are the requirements?

2. How can we successfully design and implement our tool?

3. What is an easy to use tool capable of supporting children?

4. What is information search?

As previously explained, answering the last three questions will implicitly answer the first. Thus, this chapter explains the approach to distilling information from IBM and the resulting answers to these questions.

A.2.1 Research Questions

We have broken down the research questions into the following smaller, more manageable questions, which, when answered will implicitly answer the questions above:

Requirements

1. Who determines the requirements (our stakeholders or just IBM)?

Successful design and implementation

1. What are their desirable outcomes, or what aspects make an outcome desirable?

2. How can we ensure a good translation of requirements into design, and of the design into implementation?

Support and ease of use

1. What kind of operating system (Windows, Macintosh, Linux) and software (text-editors, search engines) do children use?

2. Which characteristics do children typically show when they are on Internet (e.g. distraction, impatience, read instructions)?

3. Which characteristics make a tool simple for a child to work with?

4. How can we simplify a child's search task? What are the characteristics of a simple task?

5. What barriers do children experience in their search? And how can we help them when they run into a barrier?

6. Are there functionalities on the search engines they do not use or not use it properly? How come?

Information search

1. What is their favorite site? What functionalities do they mostly use on that site?

2. What is considered information (required characteristics of the results in order to be informative, such as appropriate content, levels of reading and understanding, and developmental stage/maturity of child)? What data must children be protected from? And who determines what inappropriate data is?

A.2.2 Distillation Method

In search of answers to the questions above, we interviewed a couple of IBM employees and read some documents written by IBM. With documents from IBM's web site and the IBM w3, we tried to gain more knowledge pertaining to the questions related to ”successful design and implementation”, and about inappropriate data. Specifically, we tried to find information about the characteristics of a successful tool, about IBM's best practice procedures of design and implementation phases, and about the company policies (with respect to user privacy, what types of specific information is not allowed to be collected from the users).

The goal of the interviews is to elicit the problems of children in the use of search engines. They are semi-structured and consist of open questions. We choose for this kind of interview because it leads to more relevant information, if something isn't clear we can ask them directly for explanation, and it gives us the possibility to change the order and logic of questioning (for example, some questions are irrelevant to some interviewees). The most complete picture would be attained by asking every relevant person on the globe, however, as this is unattainable a selection of interviewees must be made. A selection of IBM employees and people working on similar projects alongside IBM, are made according to their (history of) involvement with the project or children (relevance), function (representativeness) and accessibility. After all, these people spend a lot of time with children, so they know what their capabilities and barriers are and are good sources for determining ease of use of functionalities and appropriateness of particular information for children.

A.2.3 Sources

Our sources (interviewees) were:

1. Our client contact and Corporate Community Relations Manager.

2. An IBM software designer.

3. An IBM software programmer, DB2 information management specialist. Studied computer science and has been an employee at IBM for over 6 years. He is specialized in implementation of soft and hardware at clients, and is experienced with troubleshooting of these aspects and well-aware of barriers during programming, implementation, use and maintenance.

4. Corporate Community Relations Programme Manager at IBM in the UK. IBM employee (in a representative position) involved in research with respect to children and Internet, currently involved in a project which educates carers and educators about the dangers that Internet poses to children, and which precautions can be taken, and furthermore has a long history of contributions to relevant projects. Her current project works hand in hand with a professional American non-profit organization i-SAFE ([48]) which proactively informs kids and teens about cyberspace hazards.

5. IBM w3: In its long history of software development, IBM has developed strategies and rules of thumb, based on best practices, and has documented this in a handbook which can be found online.

We ourselves established contact with the programmer and designer that we have chosen to interviewing. Both have been IBM employees for several years, were familiar with the procedures, principles, and culture of (and therefore representative for) IBM and very experienced in their field. Since both of them were involved in troubleshooting in companies that use IBM soft and hardware they were very capable of advising us. The fourth person is a contact, of another establishment of IBM, of our client and is currently engaged in a research to children and Internet, which have overlap with our project.

A.2.4 Interviews

We took the questions posed above and moulded them into open questions which could be used for interviews. Section A.2.4 describes which questions we posed to which IBM employee, and section A.2.4 describes the interview results.

Interview Questions

In order to be able to answer the research questions posed in section A.2.1, we devised some questions that need to be answered during an interview with IBM employees. The numbers in parenthesis behind each question indicate the interviewee we plan to ask: (1) Client contact, (2) Software programmer, (3) Software designer, (4) researcher of "children and Internet".

1. **Requirements sources:** Who determines the requirements (our stakeholders or just IBM)? And who decides which requirement will be implemented when two or more requirements are in conflict with each other? (1)

2. **End terms:** For IBM, What are the end terms of the product (a prototype or must it be commercially usable)? (1)

3. **Desirable outcomes:** What are the desirable outcomes of the design/implementation phase, or which aspects make an outcome of that phase desirable? (2,3)

4. **Best practices:** What methods or procedures are used for the translation of the requirements into a design, respectively design into implementation? Have you previously run into some problems during this phase, and how did you solve them? (2,3)

5. **Ease of use:** What characteristics make a tool simple for a child to work with? In other words, what do they find simple and what do they find hard to use? What characteristics of children must you specifically keep in mind when designing a tool for them? (1,4)

6. **Experience:** What problems that children run into did you found in previous projects? How did you solve these problems? (1,4)

7. **Simplicity:** Do you have any tips to simplify the difficult functionalities? (1,4)

8. **Inappropriate data:** Are there sites on the Internet that children are not allowed to find with Giggle, where they must be protected from? (1,4) Who determines what inappropriate data is? (1)

9. **Appropriate data:** What kind of information are children able to read according to their reading level and maturity level? (1,4)

10. **Tips:** Annotations, tips, etc. (1,2,3,4)

Results from Interviews

In order to be able to answer the research questions posed in section A.2.1, we devised some questions that need to be answered during an interview with IBM employees or from literary research. The numbers in parenthesis behind each question indicate the interviewee we plan to ask: (1) Client contact, (2) Programmer, (3) Designer, (4) researcher of ”children and Internet”.

1. **Requirements sources:**
   Being the client, IBM primarily determines the requirements. However, with their interests in Corporate Community Relations, all requirements posed by stakeholders should be taken into account. IBM runs many projects hand-in-hand with non-profit organizations. Because these organizations know much more about the subject, they stand in a better position to determine which requirements should be compromised in the case of conflicting requirements.

2. **End terms:**
   The end terms of the product is a tested (by children) prototype. The results of testing (and level of content of users) must be documented. The product must be capable of retrieving documents and have an appropriate interface for use by children. It is not mandatory, though appreciated, that all requirements be implemented. However, the most important functionalities must work. There exist possibilities for extension of the project by other internists.

3. **Desirable outcomes:**
   Desirable outcomes of the design phase are a completed design document (according to IBM's AMS NL Quality Management System template) and software design checklist (established by IBM's AMS NL Quality Management System) prior to continuing to implementation. Furthermore appropriate models described in Unified Modeling Language (as a result of using IBM's Rational tool) are desirable to ensure correct and qualitative description and translation of requirements into design and the implementation that follows. On a technical level, desirable nonfunctional outcomes of the final product are performance (the software tool should adequately and timely react to user input), and its closely related scalability. Furthermore, a high level of usability as determined by the user is a desirable aspect. Specifically, the following criteria have been specified:

   - All requirements must be understood;
   - Human factors must be addressed;
   - Each new function must have an accurate and complete description;
   - There must be mapping back to Software Requirement Specification (traceability);
   - The design must be feasible;
   - Product must consist of a reasonable module breakdown;
   - Interaction between database and application should be maintained by only one module, that way, changes made in database (e.g. commands) can be dealt with by adaptation of merely one model;

- Data files must be clearly specified;
- Database definition must be clear;
- Provisions for database expansion must be made;
- Provisions for logging errors must be made;
- Performance factors must be estimated;
- Database fields must be accurately defined;
- Valid error conditions must be specified;
- Error messages must be defined;
- Dependencies must be identified;
- Internal interfaces must be defined;
- Logic errors or processing omissions, particularly in the handling of exceptional conditions, must be dealt with;
- The design must provide an appropriate level of detail to support the detail design procedure;
- The modules must be designed to be reusable;
- Where applicable, existing systems should be reused;
- The module must be modifiable;
- AMS NL methodology and standards should be adhered to;
- The project's standards should be adhered to;
- Data encapsulation must be performed, as appropriate;
- If appropriate, design language statements should be used;
- The design should be made reusable for other systems;
- Mental verification must be performed for zero defects.

4. **Best practices:**
   To improve software functionality, reliability, and performance throughout the development project, IBM has developed a software tool called Rational. Rational Best Practices have evolved for nearly 20 years through collaboration with customers and partners. The distillation of lessons learned is IBM Rational Unified Process, it combines a core set of practices with optional process Plug-Ins to support projects of any size, scope, or project environment. Rational provides tools for architecture, design modeling, construction, model-driven development, architected rapid application development (RAD), component testing, and runtime analysis activities. These tools help developers maximize their productivity when building business applications, software products and systems, and embedded systems and devices. The UML-based visual modeling and design tool for architects, systems analysts, and designers who need to ensure that their specifications, architecture, and designs are clearly defined and communicated to their stakeholders.[12]

   The most frequent problem clients run into after implementation of software is performance. Especially the performance of a database driven application is very critical. On scalability aspects, databases can be extremely large. While increasing the database size during maintenance yields no problems, aspects of hardware choice and I/O calls to disk and swapping can dramatically decrease performance. One most properly foresee the necessities in hardware and software and base design and implementation decisions on these details. Furthermore, information security is in many cases an important aspect. Features of database backup and logs should be used to restore systems where applicable. Other performance issues such as table scan versus indexing of tables, reverse scanning of tables, and decisions pertaining to the size of buffer pools have been discussed with experts in the field. Furthermore, the choice of software depends heavily on the hardware we have access to and the size of the tool to be built and support it requires. Decisions on software and database utilities have been made according to available hardware and scalability of the tool to be developed. Two IBM software specialists, experienced in advising and troubleshooting IBM clients, were willing to share their expertise, experience and opinions with us while making decisions throughout relevant phases to ensure that our project would not fail on these aspects. The choices made for use of hardware, software and database applications has been run by them for approval.

5. **Ease of use:**
   According to our client contact (due to a lack of expertise in this field), the same guidelines should be followed as with other IBM projects, such as 'Safe Internet' and those in cooperation with non-profit organizations such as 'Surf op Safe' and 'ICT op school'.

   Contact with these organizations has lead to the subsequent information. According to research by the NIPO institute, children require software with which they can work independently [42]. This means the program should have an appropriate interface, help functionalities and should be easy to learn. In order to be easy to learn the user tasks should be compatible with prior knowledge [33]. That is to say, functionality and appearance should look familiar to existing tools that are often used by the children. Children learn by means of being coached, imitating, practicing and tacit learning[33]. In order to be easy to use, the software should support at least one or more of these manners of learning to synthesize knowledge in order to assure that they fully know how to use the functionalities and remember them for subsequent visits.

6. **Experience:**
   The group consisting of carers and educators often lack computer experience [42]. The expectations of their technical support while children use the computer should be minimal and children should be capable of independently using the software. Alleviating risks and barriers can be done by empowering kids with the knowledge and skills they need to safely and responsibly take control of their Internet experiences, and how to search for information more effectively and thus more precisely. Furthermore, tools to help children in their search will help them in this process. In order to be successful the tool must be capable of working alongside existing family filters.

7. **Simplicity:**
   The same guidelines should be followed as with other IBM projects, such as 'Safe Internet' and those in cooperation with non-profit organizations such as 'Surf op Safe' and 'ICT op school'. According to research completed by NIPO, children require help for finding and selecting appropriate educative software [42]. Proper interface design

can alleviate a lot of stress and burden on the user. Appropriate help functionalities in short but clear language is very helpful to the child. Furthermore, attaining the child's attention is required to overcome difficulties. An interesting interface to keep them involved is needed.

8. **Inappropriate data:**
Dangers: The anonymity of the Internet has opened up an entirely new avenue for online predators, Cyber bullies, and identity thieves. Whats more, the Internet can lead to unwanted solicitation and provide instant access to inappropriate material. 90% of 8- to 16-year-olds have viewed porn online, most while doing homework. The sad reality is that all children who have Internet access are at risk. An uninformed, naïve child is a child at high risk. Focus should be on filtering of inappropriate information and blocking undesired features (such as (pop-up)advertisements).

What inappropriate material is, is determined by the following groups of people:

- Governments and law enforcement agencies: As an example, fighting computer crime is a priority for Germanys police force (source: German Bureau of Investigation, [48]);
- Non-profit foundations such as i-SAFE America;
- Carers and educators;
- Children.

IBM has shown to rely on governments and non-profit organizations to determine what constitutes inappropriate material. In line with this it is interesting to have a look at the Child Internet Protection Act (CIPA). The U.S. Supreme Court (United States v. American Library Association, 2003) requires any public library receiving grants to filter online content. As a primary requirement, filtering systems must prevent children from accessing the following three types of online material [71]:

- **Obscene materials** are statutorily defined as depicting sexual conduct that appeals only to prurient interests, is offensive to community standards, and lacks serious literary, artistic, political, or scientific value (18 U.S.C. sec. 1460).
- **Child pornography** is statutorily defined as depicting any form of sexual conduct or lewd exhibitionism involving minors (18 U.S.C. sec. 2256).
- **Harmful to minors** includes any depiction of nudity, sexual activity, or simulated sexual activity that has no serious literary, artistic, political, or scientific value to minors (20 U.S.C. sec. 9134(7)(B)).

Secondly, other categories of inappropriate information which children should be protected from are:

- Extensive sexually explicit content such as pornography;
- Misleading domain names or ”typo squatting” yielding inappropriate solicitation as a response to innocent keyword search by using misspellings of popular childrens characters, activities, or interests, or incorrect extensions;
- Information about making explosives;

- Advocating hate or depicting violence. The number of these sites rose from 8,667 at the end of 2003 to 10,296 at the end of April 2004, showing up at a faster rate than pornography. (Source: SurfControl);
- Mean or threatening messages (15% of students have received such messages while on the Internet);
- Information about using e-mail servers to send fake messages to other people;
- Pop-up advertisements, spam (mass-mailing e-mail), and spim (mass Instant Messages) which children are more likely to explicitly pay attention to than adults, often contain inappropriate material, including adult material and hate propaganda;
- Gambling advertisements.

Furthermore, to ensure that our website does not pose a threat to the safety of the user, according to IBM policy, the following rules must be abided by:

- **User anonymity:**
  IBM's privacy policy applying to all IBM Web Sites Worldwide states that the user should be able to visit the web site without having to say who he or she is or giving personal information [11]. This requirement specifies that it must be possible to use the product without having to log in.

- **User's personal information:**
  IBM's privacy policy also states that the user may choose to give personal information at his or her own will [11]. For the design of the product this means that we have permission to ask for information from the user (a user name and password) in order to improve customer service through personalization. However, we must intend to protect personal information and quality, implementing appropriate measures and processes (such as encryption when transmitting sensitive information) in order to keep the information secure and maintain its quality. Furthermore, we must let the user know, prior to collection, how the information given will be used.
  However, it is not our intent to collect sensitive information, a requirement posed by other stakeholder groups. In the case that a user wishes to use a personalized site, we must collect a user name and password. The user will be alerted to use a fictitious name and a new password that is not otherwise used for any other purpose in order to avoid coming in contact with sensitive information. Furthermore, all user information will be encrypted during transmission.

- **Non-identifiable information:**
  Collecting non-identifiable information from visits, such as with cookies and click-stream data, to help provide better customer service is permitted according to IBM policy [11]. IBM or others on IBM's behalf may use this data to analyze trends and statistics and to help provide better customer service. This means that we are permitted to track user visits and save data on search queries, strategies and results and analyze these in order to allow the system to learn from current users, and in turn use this information to provide more precise retrieval results.

9. **Appropriate data:**
   As our client contact is no expert in this field, the answer to this question should be found by literary research and interviews with parents and teachers.

10. **Tips:**
    Our client contact has shown the preference towards testing and requirements distillation to take place in innovative schools because their students better reflect IBM's focus group with respect to their other projects. Also, in line with IBM's global practices, creating a product that will allow for use by children of multi-cultural backgrounds is a preference. As such, a product capable of dealing with English, Spanish and Dutch users (and thus queries and documents) is wished.

    Furthermore, the two IBM software specialists were able to warn us for many pitfalls and educate us which precautions to take to avoid problems at a later stage in the development cycle.

A.2.5 Results Investigation IBM

The results to the research questions posed in section A.2.1 follow below:

- **Requirements:**
  Our client contact, together with Corporate Community Relations Programme Manager at IBM in the UK (who has experience with children and internet) determine whether or not the project result is a success. This decision is made on the basis of satisfaction of the users, which is to be tested by us.

- **Successful design and implementation:**
  On aspects pertaining to software development, following IBM's best practices (completing software design checklist and use of macro templates) and properly imbedding and following their Rational methods in our design, implementation, and coding will yield qualitative result with desirable outcomes and ensure a correct translation of requirements into design, and design into implementation.

- **Support and ease of use:**
  Children require software with which they can work with independently. The program should have an appropriate (attractive) interface, help functionalities (in short but clear language) and should be easy to learn. User tasks should be compatible with prior knowledge, functionality and appearance should look familiar to existing tools often used by the children. In order to be easy to use, the software should support at least one or more of the manners of learning (coaching, imitating, practicing and tacit learning) to assure that they fully know how to use the functionalities and remember them for subsequent visits.

- **Information search:**
  Data conforming to the following categories is not considered informative and is inappropriate to children:

  – Obscene materials: depicting sexual conduct, offensive to community standards, lacks serious literary, artistic, political, or scientific value;

- Child pornography: any form of sexual conduct or lewd exhibitionism involving minors;
- Harmful to minors: any depiction of nudity, sexual activity, simulated sexual activity, or pertaining to criminal activity (e.g. making of explosives, sending fake emails, gambling) that has no serious literary, artistic, political, or scientific value to minors;
- Advocating hate or depicting violence;
- Pop-up advertisements, spam, and spim and other advertisements.

A.2.6 Conclusion Investigation IBM

This section has explained the process and results of the investigation at IBM.

Following IBM's best practices and properly imbedding and following their Rational methods in our design, implementation, and coding will yield qualitative result with desirable outcomes and ensure a correct translation of requirements into design, and design into implementation. Furthermore, information from interviews has yielded an extensive list of specifications related to inappropriateness of data. IBM's long history in soft and hardware development has indeed lead to incredible best practices. IBM software experts were very willing to sit down with us and discuss possibilities, share experiences, and warn us for pitfalls. Furthermore, IBM recognizes not being an expert in every field, and has projects with partners to ensure appropriate levels of expertise and quality. Where applicable, we have contacted their partner organizations for additional information.

Completeness of Information

Our client contact was not able to give answers to all the questions we sought, primarily because we expected information from him that did not lie in his field of expertise. He did however show us who he relied on to give appropriate and adequate answers to the questions. As such, some organizations were able to yield us more information pertaining specifically to children, and software experts were able to give us information about design and implementation principles, allowing us to make use of IBM's best practices, and share their expertise and experiences to avoid pitfalls. Conclusively, we were able to answer all the research questions posed in the beginning of this section.

Quality of Information

All IBM employees consulted have been with the company for several years and have extensive knowledge of their field of expertise. Also, IBM's w3, which can be seen as a handbook designed with years of expertise has valuable information and describes proven methods for software creation, and thereby valid, trustworthy, and verifiable. Furthermore, their best practices are sold to companies around the world and has been built up from years of experience, can be truly seen as a valuable source of information about design and implementation methodologies.

Furthermore, as IBM often works in projects together with other organizations, especially when the field is not their focus of expertise, we were able to use information from these companies to base our results on, increasing validity and verifiability. An example is the field of children and internet, about which 'Surf op Safe' and 'ICT op school' were able to yield us much more information.

However, due to IBM's globalization, not everyone we wished to interview and all the information we wished to attain was available. Examples are documentation and project members of previous similar projects at IBM, which could greatly increase the verifiability of our results. Though we were able to make contact with IBM employees oversee, we had to limit our interviews to people within the country.

Limitations of Study

IBM is a global company with employees and offices worldwide. Though they have an extensive amount of information in the form of knowledge management, some specific details (such as reports of former relevant projects) were merely not attainable to us. Furthermore, as our project only spans several months, we were not able to completely settle in and thus meet all people who could give us relevant information. We had to restrict our conclusions based on information from only a handful of people and a handbook. We did have contact with IBM employees oversees, however, due to distance this contact was limited to email and telephone rather than a personal one-to-one interview. This limits the quality and quantity of results because in a one-to-one interview someone is much more likely to give out more information.

A.3 Investigation of Carers and Educators

Section 3.2.1 in the requirements analysis document poses several questions and sub-questions which need to be answered in order to distill all the requirements that the final product must adhere to. Carers and educators, being the people who spend a lot of time with children, and most likely to know and understand children better than any other source, can be a valuable source of information. They could contribute in finding the answers to the following questions:

1. What are the requirements?
2. What is an easy to use tool to support children?
3. What is information search?

Answering the second two questions will implicitly answer the first question. This document describes the process, methods and results of an attempt to answer the questions posed above.

A.3.1 Research Questions

We have broken down the research questions into the following smaller, more manageable questions, which, when answered will implicitly answer the questions above:

Support and ease of use

1. **Tool simplicity:** Which characteristics make a tool simple for a child to work with? What kind of things do they find easy to use?

2. **Task simplification:** How can we simplify a child's search task? What are the characteristics of a simple task?

3. **Children's characteristics:** Which characteristics do children typically show when they are on Internet (e.g. distraction, impatience, read instructions)?

4. **Hard and software:**What kind of operating system (Windows, Macintosh, Linux) and software (text-editors, search engines) do children use? Have precautions been set towards safe internet (disallow cookies, web sites or pop-ups blocked)?

5. **Children's preference:** What is their favorite site?

6. **Children's capabilities:**Are there functionalities on the search engines they do not use or not use it properly? How come?

7. **Dangers:** Which dangers or problems does (searching on) Internet have? What types of documents are inappropriate by content (levels of reading and understanding, developmental stage and maturity)?

8. **Barriers:** What barriers do children experience in their search? And how can we help them when they run into barriers?

Search for information

1. **Search approach:** How do children search for information (directories or queries)(words or whole sentences)(which search engines)?

2. **Informative:** What is considered information (required characteristics of the results in order to be informative, such as appropriate content, levels of reading and understanding, and developmental stage/maturity of child)? What data must children be protected from? And who determines what inappropriate data is?

3. **Result description:** What kind of information do they look for on the Internet (lots of pictures, stories, informative texts)? What do they consider interesting information? (lots of pictures, easy-to-read or challenging, short in length, stories vs. informative)

4. **Drive:** What drives their search (education or entertainment)? What is their goal (finding specific facts, information gathering with less well-defined information objectives, or just browsing out of curiosity)?

5. **Goal:** What is their search goal (finding specific facts, information gathering with less well-defined information objectives, or just browsing out of curiosity)?

6. **Result Expectation:** What kind of result do they want, what do they expect the result of the search is (a word, paragraph, a particular type of document)? What do they do with the result (copy-paste to text-editor, email, print)?

7. **Formulation of information need:** How clearly can they formulate their information need? Are they given specific questions to answer (in articulated form with exact terms) or must they formulate their own questions? Does this pose problems?

A.3.2 Distillation Method

The goal of the interviews is to elicit the problems of children in the use of search engines. They are semi-structured and consist of open questions. This method has been chosen in order to answer the questions we want answered, but also to possibly gain more information about topics of which we have not adequately formulated interview questions. We feel this method will create a relaxed atmosphere in which we can learn much about the experiences and opinions of our interviewees.

A.3.3 Sources

The most complete picture would be attained by asking every relevant person on the globe, however, as this is unattainable a selection of interviewees must be made. A selection of teachers, parents and librarians are made according to their (history of) involvement with the project or children (relevance), function (representativeness) and accessibility. After all, these people spend a lot of time with children, have much experience with children especially in a learning environment (whether at school, home or in a library), so they know what their capabilities and barriers are and are good sources for determining ease of use of functionalities and appropriateness of particular information for children. A disadvantage of this group of interviewees is that they may be biased, as they are in close contact with the children, or a librarian or school teacher may not be cooperative because they may not be waiting for an internet search engine at all and prefer the traditional library as a source for information.

The people we chose to interview are all momentarily involved with the children of our target age, and have been for a period of at least five years, and are thus representative for the group. Specifically, our sources (interviewees) were:

1. 3 Teachers
2. 1 Librarian
3. 1 Deputy head of middle and high school
4. 5 Parents

A.3.4 Interviews

This section describes the questions used during the semi-structured interviews with carers and educators.

Interview Questions

The questions used are exactly those described in section A.3.1. Because the interviews were semi-structured, and the manner of posing the research questions have been done in a, for the interviewers, accessible form, re-use of the questions is possible without translation.

Results from Interviews

This section describes the raw data collected during interviews:

Support and ease of use

1. **Tool simplicity:**

   Children work well with things they feel comfortable with. Because almost everyone has a windows computer at home and is educated to used Microsoft Office products at school, any tool with similar features and characteristics will be intuitive for a child to work with. They must be paying attention in order to complete a task or learn anything. It must look interesting and flashy to attract their attention (source: teacher, parent).

2. **Task simplification:**

   Tasks should be broken down into several smaller steps rather than a large task. Overview of the entire course must be given to maintain orientation and the feeling that the child is moving closer to the goal, thus avoiding frustration (source: teacher).

3. **Children's characteristics:**

   Incomplete or poorly maintained sites make children aggrevated and should not be retrieved (source: teacher, school head, parent). Pop-ups and advertisements should be blocked because they distract and irritate the child (source: teachers, parents, school head). Generally, if they are given to them, they are willing to read short instructions, however, they will generally not explicitly search for help functions and much more prefer learning by trial and error (source: teacher). They seem to have a longer attention span and are more motivated to look on internet than in a library, because it is more dynamic and possibly because it is easier to take a break from the task and do something else (source: librarian). Children often play on the computer in pairs, with another friend or a sibling (source: parent).

4. **Hard and software:**

   The most generally used operating system is Microsoft Windows 2000, browser is Microsoft Internet Explorer, and software is Microsoft Office products. All children have been educated to use, and feel comfortable working with Microsoft Office tools such as Word, Excel, PowerPoint, and Paint (source: teacher, school head, librarian). All people interviewed have a computer at home with internet connection (either cable/ADSL or modem) and use a Microsoft Windows operating system version 2000 and up. Internet Explorer is the only browser installed on those computers. In all cases the children have permission to use the computer and go on internet. Parents are not able to continuously supervise their children's activities on the computer. Only one of the parents interviewed have changed the internet security settings in the browser, the others did not know it was possible but were very interested to learn more about it. None of the parents had ever used the 'History' feature to observe their children's activities (source: parents).

5. **Children's preference:**

   When they have to do a report, they usually use Google and type in the title of their report, usually a one or two word query. For entertainment their favorite site is in most cases a game directory (source: teacher, parent).

6. **Children's capabilities:**

   There are many complex functionalities they (and I) don't even know they exist (source: teacher, parent). They understand how to work with the computer a lot better than their parents (source: parents). Most of our students are better with computers than we are, which is frustrating, because if they ask us a question we usually can't help them (source: teacher). They are unable to understand the help because it has been written for adults (source: teacher). They don't use functionalities that don't look attractive or they really have to read and concentrate on instructions. Google's Advanced Search and Preferences are an example (source: librarian).

7. **Dangers:**

   Children should be brought up in an environment where they are taught about the dangers of the Internet and furthermore feel comfortable talking about it with an adult (source: teacher). The barrier between fact and fiction has become much smaller. It is hard to determine what is true on the internet. Children are not capable of determining if sources are reliable (source: teacher). A large danger that the Internet poses are harmful programs such as viruses, automatic installations, such as toolbars and programs, which meddle with hard and software (source: parents, teacher, school head). Children structurally say 'OK' to a block-up because it's the fastest way to get rid of it. However, this behavior often brings complications with it because by doing so they authorize program or toolbar installations (source: parent). Lately there has been a lot of shocking stories in the news about cyberbullying. The possibility to exchange personal information is undesired (source: parents, teachers, school head) The capability of online shopping and banking should be prevented (source: parents).

8. **Barriers:**

   When children don't know how to search, giving help reformulating their query most often solves the problem. Query expansion is the most used strategy. Surprisingly, adults are often not capable of using boolean expressions appropriately, and functionalities such as the use of quotes are not necessarily known to everyone (source: parents, teachers, librarians).

   Because they have difficulty judging what reliable information is, they need help (a suggestion is determining the information source). For example, a widely recognized brand or institution that you trust. The site should provide a privacy statement or a Terms of Use statement. Reading these statements should make it clear what will happen when you use the site (for example, being tracked or seeing advertisements) (source: teacher).

Search for information

1. **Search approach:**

   They generally all start with Google, and prefer the use of query searching because it yields a result quickly. They try to type in as many words as they know about the search topic (source: teacher).

2. **Informative:**

   The highest priority is child safety and higher precision; preference to filtering inappropriate documents even if this leads to smaller and censored results list (source: teacher). Filtering of adult content (sexually explicit graphic descriptions or images) is necessary (source: parent and teacher). Filtering of hate sites (advocating bigotry or hatred) should be done (source: teacher). Graphic violence (such as violent images, language, bomb-building, etc.) can be harmful to a child (source: parent and teacher). Information promoting illegal or criminal activities should be withheld from children (source: teacher and school head). Other offensive sites should be filtered (with excessive use of alcohol or inappropriate games, advertising, politics, sports, etc.) (source: teacher).

3. **Result description:**

   They usually look for informative and factual documents containing lots of pictures (source: parent, teacher). Results in a language that the child is not fluent in are useless and not considered informative (source: parent, teacher). Often the language used in the resulting document is of a very high academic level, and not well understood or useful (source: parent, teacher).

4. **Drive:**

   When they search for things on internet they usually do so in search of information for a report, information gathering. They usually do not use the internet to find the answer to a specific fact. Perhaps, that costs them relatively too much time for the amount of information it yields. They rarely browse out of curiosity without a particular goal (source: parents).

5. **Goal:**

   Generally, their goal is that of less well-defined information objectives. For educational purposes they often must gather information pertaining to a particular topic, without posing specific constraints on the exact topic content. When they seek information it is often to find out more about a particular topic in general, rather than finding the answer to a specific question or finding a specific fact.

6. **Result Expectation:**

   Due to their search goal (a general search to increase overall knowledge about a topic), they often expect a document, or rather, a set of documents pertaining to the same topic. Most often, the result is copy-pasted into a Word document. Sometimes, they print the document. When this is the case they print the document in order to get the pictures. These are then physically cut out and pasted into a report. The reason for this is that controlling layout aspects in Word (pasting a picture exactly where you want it) often poses difficulties for a child. Seldomly the document is emailed, however, not to a friend but rather to themselves so that they can use and consult the document at home; repeating the exact search at home is implausible and furthermore time consuming.

7. **Formulation of information need:**

   Most often, when they look for information for a report they are allowed to select their own topic. They however, do not have much domain knowledge and are often not capable of expanding their query for reformulation. They usually start by typing in a vague words and browse through hyperlinks until an appropriate document has been found. Such a strategy is very time-consuming, does not guarantee results, and can give rise to frustration (source: teacher, librarian).

A.3.5 Results Investigation Carers and Educators

This section provides a summary of the answers to the research questions posed in section A.3.1.

In the line of support and ease of use, a simple tool is characterized as one that is similar in appearance and functionality to existing tools that they are used to and thus comfortable with using; according to our research these are Microsoft Office products and Internet Explorer. Children are impatient, though do make effort to read short instructions. Large tasks should comprise of several small steps. Furthermore, the interface must be attractive to maintain the childs attention and give them the idea it was designed for use by children and therefor they are capable of using it. Also, many dangers have been described (please see section A.3.4 for details). Domain knowledge seems to be the largest barrier during search, however, determining the reliability of a retrieved document also poses problems.

Pertaining to the search for information, in short, the following results have been discovered. Their primary search method is query based due to the timely results. Retrieved documents should be in the childs mother tongue, be appropriate for children by reading level, and furthermore should be safe. The group interviewed seems to regard childs safety as a priority and retrieval precision over retrieval recall. For educational purposes, search is often with the purpose of information gathering pertaining to a vague topic rather than specific fact finding. Several similar documents are desired in the result. Storing of intermediate results has shown to be desirable.

A.3.6 Conclusion Investigation Carers and Educators

This section described the investigation of carers and educators and the results obtained. Several aspects caught our attention. A child needs not only a search engine which will retrieve documents which they are capable of reading and adheres to their interests, aspects pertaining to safety are a hot issue. Also, the capability of storing intermediate search results or states is desirable. Furthermore, once again, the importance of an attractive interface with specific design for children has been accentuated.

Completeness of Information

All questions posed in section A.3.1 where answered during the interviews. However, we saw that in order to obtain a complete set of answers we had to consult more people than we had originally planned, not every interviewee meddles with a childs internet activities or knows themselves what the capabilities or dangers of internet are. Especially the large range of parents and educators (backgrounds in education, occupation and culture) has yielded valuable information that we could never have obtained from pure literary research. Especially the librarian, who also gives a crash course to 7th graders on how to find information on internet was able to yield rather precise and complete answers.

Quality of Information

In our opinion the quality of the information obtained is rather high. The conclusions were derived from several interviews in which the same results had been seen. Furthermore, as one of the researchers taught the focus group a crash course on computer skills, she was able to verify the results from the interviews. However, due to time restraints, all the interviews with educators were held at the same school, which may impede quality aspects, yet to our opinion not drastically. Furthermore, the carers interviewed did span a larger group. The resulting quality of the answers to the research questions, together with the other researches posed in this document, is accurate and furthermore reproducible, objective, integer, and therefor valid.

Limitations of Study

Obviously there were limitations to the study, not every carer and educator on the globe could be interviewed, nor did all the interviewees have all the expected answers. Due to time limitations we had to restrict the interviews with educators to those of only one school. However, due to the large range of interviewees we conclude that we were able to gather a rather general perception of the ideas of the entire group in question.

A.4 Investigation of Children

In order to create an easy to use tool that supports children on the Internet in their search for information, we must adequately understand how children think and work, and what types of barriers they run into. Upon having investigated these aspects we can design methods to help alleviate their problems and frustrations in relation to this topic. As described in table 3.1, the following research questions can be answered through interviewing and observing children:

- What are the requirements?
- What is an easy to use tool capable of supporting children?
- What is information search?

As previously explained, answering the last two questions will implicitly answer the first. Thus, this chapter explains the approach to distilling information from children and the resulting answers to the last two of our research questions. Details pertaining to these questions can be found in section A.4.6.

A.4.1 Research Questions

We have broken down the research questions into some smaller, more manageable sub-questions, which, when answered will implicitly answer the questions above:

Support and ease of use:

1. Which characteristics make a tool simple for a child to work with?
2. What barriers do children experience in their search? And how can we help them when they run into barriers?
3. Which characteristics do children typically show when they are on Internet (e.g. distraction, impatience, read instructions)?

4. What kind of operating system (Windows, Macintosh, Linux) and software (text-editors, search engines) do children use?

5. What is their favorite site? What functionalities do they mostly use on that site?

6. Are there functionalities on the search engines they do not use or not use it properly? How come?

Search for information:

1. How do children search? Which search and navigation strategies do they use?

2. What is considered information (required characteristics of the results in order to be informative, such as appropriate content, levels of reading and understanding, and developmental stage/maturity of child)? What data must children be protected from? And who determines what inappropriate data is?

3. What type of information they seek(lots of pictures, stories, informative texts)? What drives their search (education or entertainment)?

4. What is their search goal (finding specific facts, information gathering with less well-defined information objectives, or just browsing out of curiosity)?

5. What do they expect the result of a search to be (a word, paragraph, a particular type of document) and what do they want to do with the result (i.e. cut-and-paste, print)?

6. How clearly can they formulate their information need (i.e. search topics given in articulated form, with exact search terms)?

A.4.2 Distillation Method

In order to determine which characteristics of existing tools should be implemented in our product, which should be adjusted, and which should be disregarded, we set up experiments and interviews to determine what our group preferences and opinions were about functional and non-functional aspects of existing tools.

Though, some comparative studies have been done by other scientists we didn't want to base our conclusions and design decisions purely on literature and felt the need to get to know our users, learning what drives and motivates them. This is the most adequate manner to determine and meet their needs, involve them in the design process and get feedback on choices we have made thus far.

The search of answers to the questions posed above was fulfilled by means of observing and interviewing children. The goal of participant observations and simultaneous interviews with children is to elicit the perception, feelings, attitudes, ideas and uses of different tools and features, and to understand how children search for information. Interviews with children has shed light on aspects pertaining to this topic (details about methodology and result can be found in section A.4.4. In addition to the interviews, some specific methods were used to distill information, which will now be described.

Ease of Use and Support

In light of ”ease of use and support”, a literary research was completed (results have been documented in section A.4.6). Also a use-case has been constructed (figure A.1) describing search strategies and use of existing functionalities which has been verified by observations of children's searches (documented in section A.4.4).

Search of Information

To distill information regarding to ”search of information” a literary research has also been completed (results can be found in section A.4.6) which is used to verify and complete information that can to light during the interviews.

A.4.3 Sources

The children we chose to observe and interview volunteered to cooperate. The children selected are average children (with respect to hobbies, school performance, amount of time they spend on computers/internet) and are therefor rather representative for their age group and the focus user group. We have used a selection of both Dutch and international (multicultural) students in our experiments which corresponds with the range in background of our users.

We aimed to attain a group of children which will be representative for our user group, with focus on multinational backgrounds (as we strive for a multilingual and international applicability), age group (children in the target range and slightly above as they may deliver critical tips and comments, and though not the target user, may use the resulting program as well), and sexes (natural ratio boy:girl is about 1:1, we strive for the same ratio as we wish our tool to be used by both boys and girls).

Because children are the focus group of the product, interviewing them has obvious advantages. They know better than anyone else what they like and don't like and can show how they work, what they can work with, and what frustrates them much better than any other source can tell us. Furthermore children tend to be very honest and will let us know if they don't like something. A great disadvantage is the fact that children are mostly not very developed, especially at the ages of our target group. That means they may not always be very objective, or able to analyze and describe their own behavior. It is for that reason that we must not only interview them, but observe them as well. We keep in mind that children may not behave as usual if we are watching them, they may try to act differently in order to please us. We try to assure them that we are not testing them or their abilities, but rather current products or the product we wish to build.

For more detailed information about the participants we refer to the sections A.4.4 and A.4.5.

A.4.4 Observations and Interviews: Use-Cases

Use-Cases

A use-case represents the functional requirements, describing how a user will interact with the application that is being designed, including the responses of the software to user actions. The collection of use-cases specify all the existing ways of using the system, and thus implicitly specifies how the system reacts and which aspects must be considered during design. By evaluation of existing comparative tools we composed an orderly diagram yielding overview of the main functioning of Giggle, see figure A.1. Whether the use-cases properly reflect reality has been tested and documented in section A.4.4.

Problem Recognition
Let's start with the problem recognition. What is the problem here? Users, in our case kids, have a goal or intention. This can be educational or entertainment. They can have a need for information, but how capable are they of describing their information need? Do they know exactly what they are looking for or is their knowledge about that specified subject inadequate. What type of solutions do they expect from the system? In order to fulfill expectations, we must understand them and take them into account.

Hardware/Software
Questions we have to ask our self at this level, about the hardware and software they have access to, are:

– Which navigator, for browsing on the web, are children using (Internet Explorer, Mozilla, Opera, Netscape)? Are they browsing text-based? Do these navigators have specific characteristics which we have to take into account?
– Which operating system are they using (Window, Apple, Linux)? This is important to know due to the different possible techniques we can use, such as Flash files, Java scripts, and different picture formats.
– Where do children browse the web, and what types of hardware do they have access to in those locations? Which screen size do they have? Schools often have older and slower computers than in homes, which must be taken into account when making decisions on levels and complexity of interactivity which may pose restrictions on the system's reactivity.

Initialization
Children come to Giggle with the goal to fulfill an information need, then how does the child know about Giggle's existence in the first place? Possibilities can be, for example, by promotion of IBM, parents or teachers who tell the children about it, or children find the site by links on other Web sites (other search engines or Web sites of clubs where the children are members of). Why use they Giggle and not another search engine? What characteristics or features does Giggle have that another site doesn't have?

Help
All together, our tool can contain plenty of functionality. How can we inform the children on how to use its functions in order to optimize effectivity and user-compliance? Can this be attained by using an intro, and if so, must there be an option for novice users to skip it? Can this be attained by placing an expectation of the different functionalities behind a help button? Will children read the help before they start?

Search Initiation
Suppose a child has already chosen Giggle to fulfill his goal or intention. He goes to Giggle by filling in the address in the address bar, or by clicking on the link on some Web page. What he sees is the home page of Giggle, but what is it showing? Which interface design requirements are posed by the users? Do they expect a title bar from

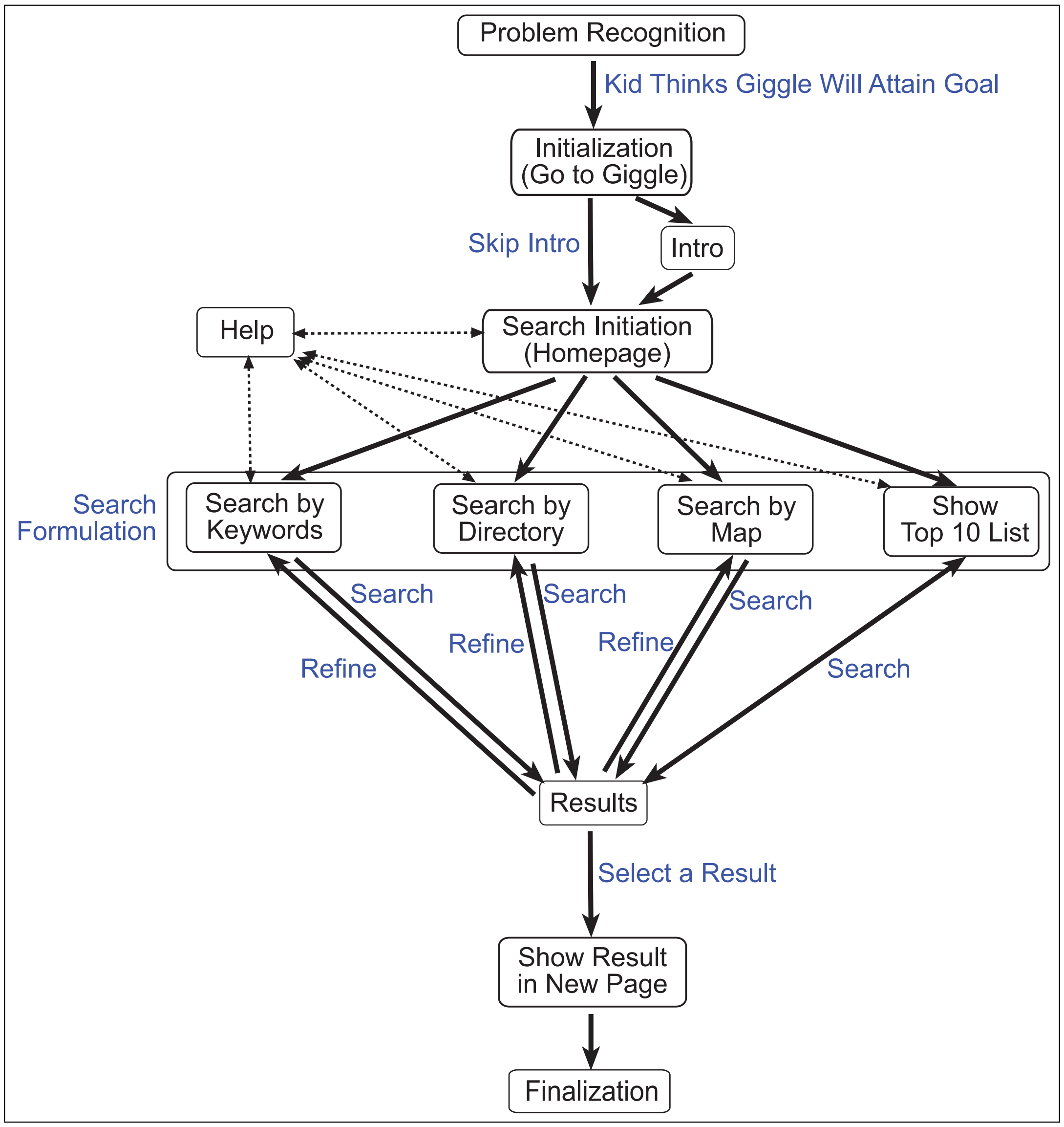


Figure A.1: Use-case diagram.

which they can select different kinds of functions? Are there specific requirements on the icons and fonts used? What colors do they like?

Besides interface requirements what functionalities do children want? Which search strategies must be offered? Search by keywords? Search by directory? Search by a map (visualization of the dependencies of different words)? Do they need help with the formulation of their search query? And if they haven't any clue where to start with the search or if they just want to look around for something without a particular goal, how can the system help? Will a top-10-search list help, where the ten most searched queries are listed and by clicking on one of them will give its results (which can later be refined)? What other functions will be used? A go back button? Go forward button? Is there a need for a dictionary to look up difficult words or synonyms? Moreover, do popups, such as advertisements, hinder or annoy our users? How important is an option to block them?

Search Formulation

Say we offer the possibility to search by typing in a query. What kind of query will a child use (single word, multiple words, perhaps complete sentences or phrases)? How familiar are they with the Boolean concepts of conjunction and disjunction (more commonly known as "AND" and "OR")?

If we implement searching in directories, which categories are useful? Must we show subcategories? How many must be presented? How can they track the path that the user chooses? What will happen when a child clicks on a (sub)category? Is it possible to quickly change from one (sub)category to another? When is the search finished, in other words, when must we show results?

The last option is to search via the relationship-map. What will be shown here? Is it a visualization representing the directory search? How many layers must be shown? And here too, when do we show results of the search? Which aspects must be updated in the interface? For example, what colors can be used and what should be made more evident? How can we represent the measure of similarity between two words (different font sizes, empirical distance, or thickness of connection lines)?

Retrieval Results

After choosing a search method and pushing the search button, a name in the relationship-map, or by making a selection in the directory, the system will start to search through documents and adhering to the selections made.

Questions that come up in this phase are: How must the results be presented? How long must the result list be? Must we highlight the occurrences within the documents of the words mentioned in the query? Which colors shall we use for emphasizing important terms? Must the source-URL be given? What items on the screen must be updated (the map, directory list, query input filed, or all items)? Will the results be shown in the same window as the home page or should a new window be opened? Will the results list permanently be visible throughout consequent searches and refinements? And while trying out other search strategies?

Search Refinement

What will happen if the search yields no results? Can the system give search suggestions or spelling variants (in case the search was too broadly c.q. specifically defined, or the child does not know how to spell a specific word)? If there are very many results, which are probably not the things he or she is looking for, can a partial ranking be given or can the child refine his search question? How? And how can we help with that?

If the users are given a list of results, is ti helpful for the user to store desired or interesting elements in a list, and remove undesired results? Is it helpful to incorporate relevance feedback? Can user relevance be used (in the form: documents deleted from list are irrelevant, documents marked useful are relevant, and indecisive about the documents not marked yet)? When a user finds some interesting documents, is an option ”more documents like this” (also as a form of relevance feedback) wanted and used by the user? Is a functionality to able to save temporary searches in order to return and complete the search at a later moment in time, seen as helpful?

Finalization
If they truly find what they where looking for, what do they wish to do with that document or information (save, print, send by mail, or copy-paste some text in another document)? Can we offer them some help with that task?

Verification of Use Cases

The goal of this first study was a preliminary research to become acquainted with our users and determine whether or not the use-cases, as we had drawn them up (figure A.1), were a correct reflection of reality. In order to ensure that this was a correct visualization we evaluated the diagram by testing it with children. Furthermore, this gave us an opportunity to speak with some of our future users and get to know how they think and work, and understand what their preferences, opinions, and capabilities are with respect to tool functionalities and differences interfaces. Several sources ([94], [63], [65], [68]) warn for pitfalls when doing experiments with children, because they think, work and are motivated in very different ways than adults, and interaction with and the questions posed to them must be done with great precaution to ensure the validity and integrity of the results. Because both of us were inexperienced in doing experiments with children, this preliminary research in which working with children on a one-to-one basis stood central, was a good opportunity to learn to work with children, talk to them, ask them questions, and keep them focussed and entertained in the meantime.

Introduction
In order to test the use-cases and user opinions and preferences of functional and non-functional characteristics of existing programs we set up a test with our focus group. Two questions were used to put the kids to work:

1. The child repeats the last search they remember they executed.
2. The child attempts to find the answer to the following question: Why don’t spiders get stuck in their own webs?
   (In Dutch: ”Waarom blijven spinnen niet aan hun eigen web kleven?”)
   Answer: Because not all parts of a spider’s web are sticky, and the spider knows where to step.

In determining the questions we looked at content and complexity levels of the questions, language use, and topic. The first is a question determined by the user himself.

The advantage of posing a topic previously searched by a child is that it gives him the capability to reflect on his experience and express what he had learned from his experience rather than merely observing what might seem like a random sequence of actions. The second is a question that has frequently been asked by children in the same age group ([46]) and is furthermore written clearly and concisely in a manner easily understood by children (this has been confirmed by several children). Because the child has not yet executed this search it allows for a setting in which the child would normally search, allowing us to observe the child's natural actions and behavior.

Research Questions

This study examined childrens information seeking on the Internet from the perspectives of cognitive and affective behaviors. The cognitive behavior relates to knowledge, comprehension, problem solving, and critical interpretation (Nahl, 1997). The affective behavior relates to feelings, perceptions, attitudes, and motivation.

This study sought answers to these questions:

1. Is the use-case a correct reflection of reality?
2. What kinds of existing tools and features do children use during their search, and why? Which specific aspects thereof do they prefer?

Method

This study employed only qualitative inquiry methods. This method generates data from interviews and observation and provides an understanding of "what", "how", and "why" children use tools. Childrens affective states and cognitive behavior were captured via one-on-one interview and observation during the study.

The setting

The study took place in Gelderland, the Netherlands. In most cases the children's own homes were the sites of the experiments, which was in most cases the primary location of their internet use and where they felt comfortable. In two cases the experiments took place in their school's computer room, the secondary location of their internet use. Several elementary and middle-school children were selected for this study. Each child's web activities, affective behaviors, and speech will be observed and those observations were recorded on paper by the us.

Participants

Our sources, thus the group children observed and interviewed, has been summarized in table A.4.4. We aimed to attain a group of children which will be representative for our user group. For a discussion about the participants, we refer to section A.4.3, Sources.

Prior knowledge of the Internet

Children had novice knowledge of using the Internet. All children had a computer with access to internet in their own home which they were allowed to use at least occasionally.

Age	Sex	Nationality
9	M	Dutch
10	M	Dutch
11	F	Dutch
11	F	Dutch
12	M	English
12	M	Colombian
12	F	American
13	F	Dutch
13	M	American
14	F	Dutch
17	F	Dutch

Table A.2: Participants of use-case testing.

The search task

All children were given the same fact-based task to research as the second question. Fact-based tasks are usually simple, certain, and uncomplicated in nature. We assigned the following fact-based task to search: "Why don't spiders get stuck in their own webs?", which can be answered in merely one sentence: "Because not all parts of a spider's web are sticky, and the spider knows where to step".

Instruments

The researchers developed a list of themes and questions to be covered during a semi-structured interview to capture the childrens affective states and motivation behind cognitive behavior. Interview data was tabulated and analyzed.

Measurements

In order to examine and compare user preferences about a particular aspect or characteristic of a tool, the user made a selection between three different icons indicating their resulting affective state. The following relevance scale was used:

1. Positive (a smiley face): an aspect which the child uses out of own initiative, knows how to use and feels comfortable with, and is furthermore confident that it will help them achieve their goal.
2. Undetermined (an 'I dunno' face): an aspect that may slightly trouble the user because he uses it but does not know exactly what it does and thereby feels slightly uncomfortable with it, yet believes it may effectively help hem to achieve his goal.
3. Negative (a frown face): an aspect that troubles the user, the user does not feel comfortable or confident using it and believes it cannot help him achieve his goal.

Procedures

This study was held in October 2004. Most of the studies took place in the homes of each of the children investigated, the primary location of their internet use and where they felt comfortable. Some also took place in their school, the secondary location of

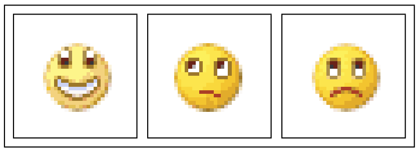


Figure A.2: Metaphores for evaluation with children: Positive, Undetermined, Negative.

internet use. The children were mainly selected on grounds for motivation to be a part of the experiments, being a member of the age group we focus on, and that they were rather representative for their age group.

Interview Questions

We distilled information from the children by means of one-to-one, semi-structured interviews. The following section describes the questions posed, and section A.4.4 documents the results.

During the case test/semi-structured interview we asked the child to perform two searches: (1) repetition the last search he has executed, and (2) attempt to find a document answering the question: Why don't spiders get stuck in their own webs? (In Dutch: "Waarom blijven spinnen niet aan hun eigen web kleven?"). Throughout the interview and observations, some topics and open questions, inline with the use-cases previously described, to be answered are:

1. **Problem Recognition:** What usually drives or motivates you to search (education or entertainment)? What is usually your goal (finding specific facts, information gathering with less well-defined information objectives, or just browsing out of curiosity)?

2. **Hard/software:** Do you usually internet at home or somewhere else? Which operating system and browser do you use and why that one and not another one?

3. **Initialization:** Which website do you usually start, and why there? Are there any things of other sites you don't like or find annoying? Do you have any ideas for new functionalities that should be added on to your favorite site?

4. **Help:** Have you ever had a look at help or intros? Did you find that useful? Why or why not?

5. **Search Initiation:** Which search and navigation strategies do you use (queries, directories, relationship maps)? Where do you usually start, and why there?

6. **Search formulation:** Are you usually given a very specific topic to look for (a teacher gives you exact terms) or is it vague and do you have to try to formulate your own question? Is that ever a problem? What do you do if you don't know what words to use? How many keywords do you type in? Do you use Boolean Keywords? Do you use quotes around phrases? Full sentence input? Would examples (top-10-list) of things that other kids look for help you get started?

7. **Retrieval Results:** Do you only select the top item in a results list? Do you read the article summary? Do you look at the URL? What kind of result do you want or expect (a word, paragraph, a particular type of document)? What does it have to have to be interesting to do (many pictures, short, easy-to-read, challenging to read)? Are there types of results you don't want, find harmful or annoying?

8. **Search refinement:** When your first attempt doesn't give you what you want, what do you try to do? Do you ever go back to the results list?

9. **Finalization:** What do you do with the results? Copy-paste, read, save, print, send per email?

10. **Feature evaluation using Table A.3 [1] :** Which features do/don't you like and why? What do you r*egularly use? When, or for what purpose, do you use it? Would you have used it if you knew about it?

Using some existing tools as examples, the child's opinion and preference over some functionalities can be asked. The intention of these questions is for the researchers to determine whether an existing feature should be kept, adjusted or removed. The child can indicate their opinion about the feature by choosing one of the three metaphor smileys from figure A.4.4; the researcher is expected to ask and jot down what the primary reason is for that particular judgement (See Table A.3).

Results from Interviews and Observations

This section mentions all the results obtained during use-case testing and interviews:

1. **Problem Recognition:**

   Most prominent driving motivation is entertainment, secondary is education. Goals are (ordered in frequency):

   (a) information gathering with less well-defined information objectives
   (b) browsing out of curiosity
   (c) specific fact finding

   Topics most frequently searched (in order of frequency):

   (a) Games
   (b) Personal interest (favorite music artist, sport, images)
   (c) School work

2. **Hard/software:**

   Location: Usually at home, sometimes at a friend's house, occasionally at school, never in the public library or an internet cafe.

[1] Links to example webpages containing features can be found at http://www.student.kun.nl/rweeda/frame_giggle.htm

Id	Feature	Score			Example
a	Category listings	😃	😐	☹	KidsClick!
		😃	😐	☹	Dogpile
b	Relationship-map	😃	😐	☹	AquaBrowser
		😃	😐	☹	KartOO
c	Results list	😃	😐	☹	LookSmart
		😃	😐	☹	Highway 61
		😃	😐	☹	Ask Jeeves for Kids
d	Help function	😃	😐	☹	KidsClick!
		😃	😐	☹	Google
e	Advanced search	😃	😐	☹	Google
		😃	😐	☹	Yahoo
f	Personalization	😃	😐	☹	Google
		😃	😐	☹	Dogpile
g	Animation + interface (metaphor, colors)	😃	😐	☹	Yahooligans
		😃	😐	☹	Ask Jeeves for Kids
		😃	😐	☹	LycosZone
h	Entertainment (games, jokes)	😃	😐	☹	Ask Jeeves for Kids
		😃	😐	☹	Yahooligans!
i	Calculator (evaluate mathematical expressions)	😃	😐	☹	Google
j	Similar Pages (display more documents like result)	😃	😐	☹	Google
		😃	😐	☹	KartOO (LIKE:)
k	Spell checker (check query input, offer alternatives)	😃	😐	☹	Google
l	Definitions (online glossary definitions)	😃	😐	☹	Google (define:)
m	Page translation	😃	😐	☹	AltaVista
n	Informative links (dictionary, encyclopedia, search engines)	😃	😐	☹	Ask Jeeves for Kids

Table A.3: Questionnaire for feature evaluation by children

Technical restrictions: Cookies are sometimes disallowed on home computers. School computers are rebooted daily resulting in loss of all saved cookies. School computers do not allow the installation of programs by the user. The home computers of some, yet less than half, of the interviewees have the same restriction.

Operating System: The only operating system used is Microsoft Windows (98, 2000 and XP).

Browser: Internet Explorer, but Mozilla is also rarely used.

3. **Initialization:**

   Interesting is that most children will try to use the first page they see to seek information. They will use the starting page first, whichever it is, after which if no results are retrieved, they will go to their favorite search engine.

   For the large majority Google is the preferred search engine. Reason for this is because it is well known, easy to use and always retrieves at least something. The second most popular search site is startpagina.nl, a directory. Sometimes Ask Jeeves, Altavista, and Yahoo are also used, depending on the type of search and the quality of the results yielded by other search engines. When the result is less specific (a game rather than having the name of specific game) the search starts in a directory.

   Suggestions by children:

   - Open all new results on a new screen.
   - Retrieve only results to a language fluent to the user, or translate it.
   - Spelling check on query input.
   - Allow for personalization such as choosing background colors or metaphors (like the Microsoft paperclip or cat).
   - A bookmark list of favorite links.
   - A list of fun links (eg. games, jokes, and quizes).
   - A list of helpful informative links (other search engines, dictionaries, encyclopedias, etc.)
   - Show the current date

4. **Help:**

   Children seem to prefer to click around and discover functionalities rather than look through help. The main reason for this is because the help hasn't been written specifically for them; in uncomprehensible language and an colorless unattractive interface with lots of (lengthy) information crammed into a page and scarcely pictures. However, when they do use the help (as with games), they tend to take the time to read it properly and follow instructions. This type of trial and error is usually reversible, it usually doesn't do any harm or lead to confusing errors, yet is a fun and interactive manner to learn what functionalities are available.

5. **Search Initiation:**

   Children primarily use query input for well-defined information objectives. Sometimes use of directory is made when the topic is a high level classification and no specific terms are known (example: goal to download a game when no game name is known), or when just browsing out of curiosity, otherwise they find it hard to find the right information. Relationship maps are never used because interviewees have never encountered them. Now they know about its existence, they find this easy to use and very convenient.

6. **Search formulation:**

   Query input:
   When the search topic is given they try to extract the most important terms (between 1 and 3 terms) and type those in. They limit the use of words to input because typing is rather slow and they often make typos which costs them too much time. Full sentences (with or without punctuation) are not used. Generally, multiple keywords are used. The children note that single keywords often yield results too general for their information need and have to browse through too many results. Quotes are not used because they are merely not acquainted with its use and semantics. Boolean keywords are never used. Children are not able to comprehend nor syntax nor its semantics. They are unable to understand the results of a boolean based query. In general, when a child wants to perform an OR search, they execute the search twice by hand. The results yielded relate more to their expectations giving them a stronger feeling of control.

   Query formulation:
   Problems arise when children must formulate their own questions because they often only have one term in mind, but are not quite sure which other words to use. Some use a dictionary to find other search words. Top-10-list will only help if it is about their search topic.

   Children seem very impatient. There is barely any thought in the process of which words to type in, in other words, they do not relate the query formulation to the expected result, but merely only to the question they must find an answer to. If their initial query seems inappropriate they are quick to try something else.

7. **Retrieval Results:**

   Results list:
   The top item is not always selected. The title and summary (with highlighted query terms) is always read and the primary source to determining whether a document may be relevant. The URL in the results list is not looked at. Scrolling to the bottom of the first page rarely occurs, and subsequent result pages are not looked at. At most they tend to glance at no more than 6 of the result summaries. If those first results don't adhere, the interviewees tend to prefer query reformulation than continue going over the list. They only bother to look at other pages if they dont have any clue how to reformulate the query, what words can be used to search on.

   If the summary doesn't quickly give them insight into the document they click on the link. The decision whether the document is relevant or not is made within seconds. Layout (interface attractiveness) and title are primary measures influencing this choice. The Google in-cache functionality shows retrieved documents with the query terms highlighted; this functionality is often used solely for that purpose. If the document seems irrelevant, the user always returns to the previous results list (using the 'Back' button) because it holds the last query typed in.

Expectations:
When searching for games the child prefers a list of games in the form of a directory, sorted by topic/types of games. Links to several directories are preferred (a similar pages function) to merely one directory. Also, because this exact same search is often repeated, being able to save the solutions found in a list is preferred.

When search is for educational purposes we see a distinction between two main types of searches:

(a) a specific topic is given along with exact terms to search on (interviewees just type in all terms given) and expect (or prefer) short specific results.

(b) children determine, often with the help of a teacher or parent, a research topic of their choice. This is often a one-word wide category such as 'Egypt', or 'Lion'. The children are stimulated to find a specific focus of their choice. Once the search is initiated the child may find that 'Egyptian Pyramids' is a more interesting topic than merely 'Egypt'. In this type of search, the child expects several general documents with similar types of results.

Search for homework help out of own initiative is not done because the interviewees mention not knowing where to look.

Favorable:
In order for a document to be favorable, it should be written in a language (level) comprehensible to the child, in their mother tongue (i.e. Dutch children are capable of reading a few words of English but are not capable of comprehending an entire English document). Furthermore, it should have some pictures. Requirements posed to interface of the search engine (attractiveness and colors) are not required in the resulting documents. Rather, a more serious interface is expected or else the contents validity is questioned.

Undesirable:
Undesirable are untargetted banners, pop-ups, payed advertisements, hate sites, pornographic material, poorly maintained sites (such as having links to sites that don't exist anymore). Also results in a type of language the child cannot read or written for a reading level beyond their comprehension are undesirable. Sites marked as being undesirable by one child will most probably be undesirable for all children. Furthermore undesirable are sites that require login to use any functionality at all. It should be possible to use sites even without login, with only extended functionalities restricted to members.

8. **Search refinement:**

During results refinement, the initial response is the removal of terms from the query, irrespective if there were no results or results not adhering to what was expected. Second response is the addition of one or more terms (related to the question) to the query. A third response is using synonyms to replace an original term (the query is not expanded). Most often the synonym was not known beforehand, but found from a relevant (but unfavorable) document which was retrieved from the initial search formulation. Sometimes a dictionary was used to find a synonymous term. If the results still seem too general (as a result of using a general query) additional keywords are usually not added; rather, a whole new query is typed in.

Kidsclick! deletes the input query when no results are shown, interviewees have noted this is annoying because it does not allow for query refinement.

When browsing out of curiosity, the children seem to be more patient. Often, an initial query is typed in (sometimes followed by a refinement) whereafter they prominently follow hyperlinks in search of related information. This, however is not the same as similar pages. They are not looking for more of the same information, but rather more information pertaining to the same topic. (E.g. a search for a specific Audi car does not mean the child wants information about other models or brands, but often rather wants to know which accessories can be bought for the car.) They thus prefer documents containing hyponyms and meronyms above hypernyms and synonyms.

9. **Finalization:** The results are most often cut-and-paste or manually typed over into a Word document. Entire web pages are also sometimes printed, especially for the pictures, and some pages are saved onto disk (usually harddisk at home and floppy when at school). They rarely send links of interesting sites via instant messenger or email to their friends, but when they do, they never send actual page contents. Children are often limited to the amount of time they are allowed to spend on the computer and end up frequently redoing their searches. They are often only permitted to use the computer for a certain time, and if no results are obtained to save or print, they re-initiate the exact same search at a later point in time.For this reason their search is limited in time, and the capability to save intermediate results and search state would be profitable.

10. **Feature evaluation using Table A.3:**

    (a) Category listings: Is useful if the search is vague and high level (only known term is a category name) and the child doesn't know how to adequately specify a query. This function is often used in combination with query search; first a category is selected to narrow down results, whereafter a query is input with the expectation that it restricts itself to the category selected. Most often this is because due to an initial lack of domain knowledge. The number of categories used in KidsClick! was perceived as being too many, making a selection is very difficult; a child doesn't want to have to read a whole page full of words before being able to make a selection. Category pages often used are startkabel.nl and startpagina.nl.

    (b) Relationship-map: When introduced to a relationship map, the interviewees seem to grasp its concept very quickly, find it intuitive and enjoyable to use and seem rather efficient and effective at localizing relevant information. It allows for quick selection of terms with fast results. However, upon selection of a document, Aquabrowser first opens a summary with extra information about the website. This is rather confusing to the child. Immediate transportation to that website it preferred and expected.

    (c) Results list: Little differences in results lists are noticed. A short results list is preferred. Also, the page summary with highlighted keywords is liked, however it should not be too long. The URL is never used. The functionality 'similar pages' is not used because its functionality and results are not understood.

    (d) Help function: Most interviewees almost never read the help, they trying things out. One interviewee said he always skipped introductions because he found them annoying. Three others said they found them useful, but only watched them if they were attractive and flashy.

Most help functions have been written for adults, in lengthy complicated language and with an unattractive interface. This turns children off, making them click it away without even trying to find an answer to their question. If they don't feel it has been written specifically for them, they won't read it.

When playing games, children do have the patience to read the instructions. Reasons for this is that they can't proceed to play if they don't know which keys to use, and the instructions have been written in a language clear to them.

Separate help functions are rarely used because it often takes long to find the answer to a question; the texts are too long, hard to understand, and often answer everything except what the child wants to know. The children prefer learning by trial and error. Though often found annoying, the Microsoft paperclip is liked because "he's funny" and gives short, easy-to-read tips in a small dosages.

(e) Advanced search: Children are not capable of understanding neither the input of Boolean queries nor its retrieval results. Even separate input fields to aid in the input of Boolean queries (such as with Google's Advanced Search) is not understood. Other features in advanced search, such as search in an URL or in a specific domein, are also never used due to unfamiliarity and complexity.

(f) Personalization: Only one interviewee used this function, to open links on a separate page. Another one used this ones to set the language of the retrieval results. Others were scared off by its difficult language and boring appearance.

(g) Animation and interface (metaphor, colors): Very important! A fun and colorful interface makes a child relaxed and his visit more pleasurable. Especially the Yahooligans interface was found very attractive. The date is nice to have on the page! If a site doesn't look like it has been made for kids, such as Yahoo or Lycos, they won't use it. The word 'kids' must be included in the title or another prominent place to attract attention.

(h) Entertainment (games or jokes): Very important! Children prefer fun interfaces with lots of links to games. Their favorite game sites are www.newsgrounds.com, www.candystand.com, and www.spele.nl.

(i) Calculator (evaluate mathematical expressions): Children are rather enthusiastic about the functionality by doubt ever using it. Calculations while sitting behind the computer are rarely or never made.

(j) Similar pages (display more documents like results): Children have never used this functionality. They don't know what the button means (complex language) and have no idea what it does. The resulting page also looks different than the standard results page, confusing the child.

Children seem to be capable of focussing on one thing at a time. During search of an initial document, they cannot foresee needing more of similar information but are focussed on their query formulation. Possibly, such a functionality should only be shown, and at that time, emphasized, when the child has found a relevant document. When shown in an example, the child found it a very useful functionality.

(k) Spell checker (check query input and offer alternatives): Most interviewees were very happy with this function and used it very often. One interviewee did not understand why or what it was because there was no explanation indicating that the input spelling was/may be incorrect.

(l) Definitions (find glossary definition from other online sources): The functionality was nice. However, yielding a intermediate list of links was not expected, rather a direct definition should be giving.

(m) Page translation: To children fluent in English this isn't a very interesting functionality because most pages are in English and an automatic translation will never yield an easy to read result. However, Dutch children not fluent in English run into the problem of not finding sufficient information in Dutch, while there are numerous sources in English, and would definitely be helped with such a functionality.

They were very enthusiastic about babelfish, the translation page of AltaVista, where complete pages can be translated.

It remains to say that an automatic translation is never faultless due to the ambiguity of natural language, and especially when a lot of pictures is used to build up the page for which text extraction is a barrier.

(n) Informative links (dictionary, encyclopedia, other search engines): Children have mentioned to use (bilingual) dictionaries and encyclopedias a lot. The Yahoo functionality 'definition:' which yields a dictionary definition of the word following it is often used. Homework help (like www.purplemath.com) and example presentations and reports are useful. Tools such as a translator (www.worldlingo.com), an encyclopedia (encarta.msn.com, wikipedia.org) and an atlas are often consulted.

Limitations of Use-Case Study

The intent of this investigation was primarily to test whether the use-case diagram reflects reality. The specific aspects of query terms input, effectiveness, efficiency and quality of search were not taken into account at this point in time; only the method and approach to searching are interesting aspects. Because of emphasis of this investigation, the language in which the search was conducted was not extremely important. Because children fluent in English are hard to come by in the Netherlands, we therefor primarily used Dutch children to test the use-case, though we understand that this decision may bias our results because the children researched may not reflect the international community that comprises the user group of the final product. In order to restrict this limitation we included four international students.

Conclusion Verification Use-Cases

The results are reported within the context of the two research questions posed. The results were based on 11 childrens Internet sessions and open interviews.

This study sought answers to these questions:

1. *Is the use-case a correct reflection of reality?*

2. *What kinds of existing tools and features do children use during their search, and why? Which specific aspects thereof do they prefer?*

   The cognitive and affective behavior of all children was observed. Furthermore, children indicated how pleased they were with different functionalities they ran into.

As a result of our observations and interviews, the use-case seems to be a rather correct reflection of reality. The only aspect that caught our attention to be very different is the use of introductions and help. Children merely do not use this functionality. Furthermore, what we cannot capture in the model is the fact that children often initiate a search and break it off, only to continue at another time, during which they initiate the same search over again.

As an answer to the second question, we can conclude that children most often execute directory and query based searches. However, a relationship map is intuitive, fun and simplifies the search process by posing relative terms to query input, thereby alleviating the child's primary barrier, that of domain knowledge. Children are all rather capable of working with Microsoft Office products and feel comfortable using the web, there is no need to expect problems with the use of common functionalities similar to that of these products.

A.4.5 Observations: Usability of Existing Tools

Usability of Existing Tools

The second experiment we set up had a slightly different emphasis than the first. Though the goal was also to determine which aspects of existing tools should be implemented, adjusted or disposed of in the final product, the approach and focus lay elsewhere. Where the first experiment concentrated on what children use, why they use it and what their preferences are, this experiment focussed on aspects of efficiency, effectiveness and quality of searches with respect to different search strategies used. In an attempt to improve their searches with our new tool, this data will reflect not only what children prefer, but rather what works best for them, giving us statistical grounds on which we can base design decisions.

Because the first experiment was done on Dutch children, and for the reasons explained previously, may not reflect the international community, we can use the data in this experiment to validate the results of the previous experiment.

Introduction
In order to test the usability of existing programs we set up a test with our focus group. They were to search for answers to two different questions, a closed and an open question.

1. Do your eyes grow as you get older?
   Answer: Yes, our eyes grow as we grow and usually stop in our late teens.
2. Why don't spiders get stuck in their own webs?
   Answer: Because not all parts of a spider's web are sticky, and the spider knows where to step.

In determining the questions we looked at content and complexity levels of the questions, language use, and topic. Both questions had frequently been asked by children in the same age group ([46]) and are furthermore written clearly and concisely in a manner easily understood by children (this has been confirmed by several children).

Research Questions
This study examined childrens information seeking on the Internet from three perspectives: cognitive, physical, and affective behaviors. The cognitive behavior relates to knowledge, comprehension, problem solving, and critical interpretation (Nahl, 1997). The physical behavior concerns actions made other than searching and browsing, such as screen scrolling, activating browser command features (e.g., Back command),

target location and deviation, and time taken to complete the task. The affective behavior relates to feelings, perceptions, attitudes, and motivation.

This study sought answers to these questions:

1. How successful are children in finding the correct answer to a fact-finding task?
2. What kinds of cognitive behavior do children demonstrate during the task?
3. What kinds of physical behaviors do children demonstrate during the task?
4. What differences in weighted traversal effectiveness, efficiency, and quality Web moves does the Web Traversal Measure reveal between children using different search strategies?
5. What affective behaviors do children experience during the task?

Method
This study employed both quantitative and qualitative inquiry methods. The quantitative method provides empirical data about the behavior, success, problem solving, Web navigation skills, and knowledge of the search engine used. Because this method requires that observations be recorded and viewed at a later time, Bulent's Screen Recorder (software that records and replays captured activities in Web browsers) was installed on all the computers. The Screen Recorder was configured to make 40 shots a minute, yielding files of about 300MB per session. By completing some preliminary tests, we deduced that the quality and frequency of the shots was good enough to determine which keywords are used, which links are selected, whether or not scrolling is used and generally which moves are made with the mouse. The qualitative method generates data from interviews and observations, and provides an understanding of the behavior of the data that result from the quantitative method. Childrens affective states were captured through observations during the study.

The setting
This study took place at an International middle school, IS1, IS2 and IS3 grades (comparative to 7th, 8th and 9th grade), located in Arnhem, the Netherlands. The schools media center is the site for this experiment. Three classes were selected for this study.

Bulent's Screen Recorder was installed on each of the computers and pre-tested for proper operation. The Microsoft Internet Explorer browser was used and the school's web site was set up as the default home page in the browser. Each childs Web activities were captured, saved, and transferred electronically to the researchers computers.

Participants
The students involved in the study comprised of:

- 14 7th graders
- 12 8th graders
- 12 9th graders

All students had international backgrounds, either originally coming from another country or having been brought up oversees. About half of the children of each age group was boy, the other half was girl. For a discussion about the choices of the participant group and the advantages and disadvantages of these interviewees we refer to section A.4.3, Sources.

Prior knowledge of the Internet
Children had novice knowledge of using the Internet. The Middle School previously offered a course on computers and some of its tools to the students, in which they to use of the Internet was mandatory. Furthermore, due in part to the Middle Years Programme technology programme, teachers regularly encourage the use of Internet as a source of information.

The search task
All groups were given the same fact-based task. Fact-based tasks are usually simple, certain, and uncomplicated in nature with an answer such as a number. We assigned the following fact-based task to search: "Do your eyes grow as you get older?". This choice was made because well-documented experiments on the same focus group have used a similar task, allowing for easy comparison of results.

In order to test searching on an open question for which we also chose the following question: "Why don't spiders get stuck in their own webs?".

Measurements
A Web Traversal Measure developed by [59] examines weighted traversal effectiveness, efficiency, and quality of Web moves. The measure is based on a weight that is assigned to every Web move a user makes. Traversal in this measure is defined as all moves a user makes including searching, browsing, screen scrolling, backtracking (using the Back button), search looping (re-execution of searches previously made), and hyperlink looping (re-activation of hyperlinks previously visited).

In the following equations, the Transcribed Moves (TMs) are all traversal moves, and Selection Actions (SAs) are all the moves that include only searching and/or hyperlink activation and weight (WSA) is assigned to each SA based on its degree of relevance.

- Effectiveness is evaluated in terms of the amount of effort a user makes to locate a target hyperlink or page.

$$\text{Weighted effectiveness score} = \frac{\sum_{i=1}^{i=n}(WSA_i * SA_i)}{\sum_{j=1}^{j=m}(TM_j)} \tag{A.1}$$

  where n is the total number of SAs and m is the total number of TMs to the target hyperlink.

- Efficiency is assessed based on the weight of relevant moves a user makes out of the total traversal moves to complete the task.

$$\text{Weighted efficiency score} = \frac{\sum_{i=1}^{i=n}(WSA_i)}{\sum_{j=1}^{j=k}(TM_j)} \tag{A.2}$$

  where n is the total number of SAs and k is the total number of all TMs.

- Quality moves are computed by quantifying the percentage of relevant moves a user makes out of the total traversal moves (i.e., quality vs. quantity).

$$\text{Quality moves} = \frac{\sum_{i=1}^{i=n}(SA_i)}{\sum_{j=1}^{j=k}(TM_j)} \tag{A.3}$$

  where n is the total number of SAs and k is the totalnumber of all TMs.

Because both the titles of hyperlinks and their descriptions may be either accurate, misleading or vague, a relevance scale (WSA as defined by Bilal [59]) was used to describe search move or hyperlink relevance:

- 1 = Relevant: a search move or hyperlink which, based on its formulation and/or description, is appropriate or appears to lead to the desired information and it does.

  Search example: spider

  Hyperlink example: Science and Oddities:Living Things:Animals

- 0.5 = Semi-relevant: a search or hyperlink which, based on its formulation and/or description, is appropriate or appears to lead to the desired information but it does not.

  Search example: spider web

  Hyperlink example: Charlotte's web

- 0 = Irrelevant: a search or hyperlink which, based on its formulation and/or description gives no indication of, and does not contain information relating to the search task.

  Search example: spider web

  Hyperlink example: Animals

Success measure

Success rates are easy to understand and represent usability's bottom line. We evaluated the search results that the children submitted. Both student groups were judged to be successful if they found and extracted the correct fact (i.e., age of alligator in the wild and in captivity). They were judged to be partially successful if they submitted an incomplete answer (i.e., age of alligator in the wild or in captivity). They were judged to be unsuccessful if they submitted an incorrect answer.

Procedures

The study was held in November 2004. The study took place in an International Middle School located in Arnhem, the Netherlands. The School was mainly selected for its pioneering efforts in integrating the use of technology into the curriculum, by means of the International Baccalaureate Organization's Middle Years Programme ([38]). The prime focus of this program is that learning how to learn and how to evaluate information critically is as important as the content of the disciplines that are taught. Furthermore, the school was chosen because of the large diversity of international backgrounds and that all students are capable of speaking, reading and writing English fluently.

The research experiment took place in the schools media center. Children were tested a class at a time, on average 12 students. Each child had access to an own computer station that have the school's home page set up as the default Web page. Children were not given explicit instructions as to how to search for information on the Web, the intent was to examine how they used Web tools to support their information seeking.

Each student was assigned 6 minutes to complete each sub-task. When technical problems occurred, this was noted. Each students Internet session was recorded by a Screen recorder. At completion, each students Web session was saved. It was planned to transfer these saved sessions electronically to the researchers computers, after which each session could be replayed, analyzed, transcribed, and coded by the researchers. However, due to restrictions on the computers (no capability to install a USB-pen, no capability to adjust IP-numbers for transfer by UTP cable, no capability to download programs for FTP transfer, no capability to access webmail), we were unable to transfer the sessions to our own computers and had to evaluate at that particular time because all files are deleted upon computer reboot. The screen recorder and all data was removed from the computers by rebooting them.

Following the sessions, each child was instructed to evaluate their search by filling out a questionnaire, made by us, on Internet. The results of the questionnaire were emailed to us automatically including information about how long each student spent searching and answering the questions respectively. Afterwards we evaluated the search session as a class with all the children. We allowed room for a class discussion so that children could explain to us and each other how they attempted their search, the problems they ran into, and what they would do differently in the future. The children were given a short break during which they were be given some lemonade to drink. This allowed us to determine aspects dealing with affective behaviors with respect to the task. Following this, the we gave an explanation on how to use search engines and gave the children tips on how to improve their future searches. Afterwards some explanation was given on the dangers of internet and steps children can take to ensure their safety while they are online. The children were thanked for their cooperation. After class children who wished were free to comment on our prototype or screen shots as they were at that present time.

Results Usability Existing Tools

The results are reported within the context of the five research questions posed. The results were based on 36 of childrens Internet sessions. Where results were incomplete they were disregarded from the experimentation.

This study sought answers to these questions:

1. *How successful are children in finding the correct answer to a fact-finding task?*

Tool	Successful	Unsuccessful
Query Input	10	6
Map	5	3
Directory	0	9

According to the results (percentage wise), children were just as successful using the map as the query input. The directory showed dramatically appalling results, no student was able to find the correct answer.

2. *What kinds of cognitive behavior do children demonstrate during the task?*

   The cognitive behavior of the students were observed in terms of searching and browsing moves.

   Searching moves:
   Query inputs:

Question 1	Question 2
eye growth	Why don't spiders stick to their webs
Do your eyes grow as you get older	spider webs
growth of eyes	Why don't spiders get stuck in their own webs?
Do eyes grow as you get older?	why not spider stuck in web
eye	spiders
grow eyes older	spder
do my eyes grow	spiders not getting stuck in web
eyegrowth	spider webs stuck
growth	

   About 10% of the searches made use of single concepts, 5% using two concepts, 70% using phrases or natural language, and the remaining used 3 concepts but not phrases. Spelling mistakes made in query input field, however immediately noticed and corrected by the user. Children showed to have extreme problems in the reformulation of a query (morphological forms, synonyms) which lead to great frustration.

   Browsing moves:
   All users made the same category selection path:
   Science and math ⇒ Animals ⇒ Spiders and scorpions
   Conclusively, all children activated appropriate categories and hyperlinks. Children indicated preferring to search by keyword than by directory. When given a free choice 100% of the children choose for query input.

   Children using the directory also often made use of the query input field. The input queries have been mentioned above.

   In addition, it was noted that children looped searches (re-executed searches previously made) and hyperlinks (re-activated hyperlinks previously visited) when the original results were unsatisfactory. Given the fact that the Web imposes memory overload that reduces recall during navigation (Cockburn & Jones, 1996), children become prone to loop searches and hyperlinks more frequently than adults.

3. *What kinds of physical behaviors do children demonstrate during the task?*

The physical behavior was examined in relation to the moves that the children made other than searching and browsing. These included backtracking (use of Back button), screen scrolling, target location and deviation, and the time taken to complete the task.

When successful, all students completed their search within 4 minutes. Those that were unsuccessful noted that they would never find the answer using that particular search engine.

4. *What differences in weighted traversal effectiveness, efficiency, and quality Web moves does the Web Traversal Measure reveal between children using different search strategies?*

   Those unsuccessful using query input were so because they 1) typed in the entire question which did not lead to any results, or 2) typed in a very general term which lead to many irrelevant results in which the children were not capable of finding specific answers.

5. *What affective behaviors do children experience during the task?*

   When results were posed that had nothing to do with the input query the users became agitated and frustrated. While using the directory, one of the children even remarked: "I didn't find the answer, because when clicked on "spider" i got a lot of junk about scorpions and horses, which i certainly didn't need. There wasn't much information on spiders. Would it just be easier to type "horses" when I need information on spiders??" Many of the results obtained by query irrelevant, and yielded far too many results.

   The opposite was also true. Using the directory the children complained that no results were yielded if more than one term was input in the query input field.

   Children note that query input is easy and intuitive to use.

   They found the layout of the relationship map confusing because they didn't know where to find the information pertaining to the document. It was rather difficult to understand and manage. However, the children note experiencing a steep learning curve. The map also only yielded one answer, which was exactly what they searched for, yet in most cases not appropriate to their information need because they often wish more information pertaining to a subject, and not a specific fact. Child liked that more information was given about the topic than merely the answer to the question.

   Younger children had more troubles coming up with variant keywords or term morphs to use in the query input field.

   The directory interface and query interfaces were boring, unattractive and not colorful.

   The relationship map took a while to load, however, because of the amusing metaphor the children did not mind. The children foudn the site colorful and cool.

   What was also very interesting is that children often want more information pertaining to a topic than they may explicitly have in mind during the initiation of a search.

Limitations of Study Existing Tools

The study was limited to seventh, eighth and ninth grade students in one school. The children who participated in this study may not represent the cognitive, physical, and affective behaviors of all middle school students in the world, nor may they represent the whole population of students in their grades. However, because testing was done at an international school, the interviewed subjects give a better representation of children world-wide than if testing had been limited to a Dutch school.

Technical problems restricted the evaluation of the screen recordings. This was a shame because it may have yielded more precise results pertaining to physical search behavior. However, we did study and report their behavior and their search strategies which made evaluation of the testing possible, yet less precise, integer, and reproducible.

Conclusion Study Existing Tools

This investigation was very interesting and valuable to ensure us that we were on the right track with respect to which barriers children run into and how to alleviate these. Furthermore, spending more time with several children during their searches gave us more feeling for what their preferences were and what they use Internet for in the first place, which will be especially valuable when designing an interface and in determining which functionalities must be included in the final product.

A.4.6 Literary Research

Ease of Use and Support

In order to make a tool that is capable of supporting children, meaning that it aids children in finding the information they are looking for, it is necessary to understand the relevant problems that children run into during their search and the specific needs they have. Furthermore, only a tool that is easy to use is capable of supporting a child. It is for this reason that we answer both aspects at once. We thus want to find answers to the question: ”What is an easy to use tool capable of supporting children?” as has been defined in section 3.2.1. The best way to do this, is by observing and interviewing children and by completing a literature research. Results from observation and interviews are discussed towards the end of this section in section A.4.4. The results from literature research have been analyzed and formulated as user requirements, to be found in section 3.4.2.

An additional initiative in line with this research is a course that we volunteered to give at an international middle school. The course educates children in the use of text-editors, spreadsheets, presentation tools, but also includes a lesson about the dangers and safety precautions on the internet. The main reason we offered our services is to gain hands-on experience about how children think and work. This gave us an indication about how they read and follow instructions, what they like, what motivates and drives them, what their background knowledge is, and what they are capable of. The only way to really learn about children is to spend time with them. All of this information has been found to be extremely useful throughout this research.

Internet Use

To commence, we inventoried which tasks children use the internet for. Because it is difficult to find statistics about, among others, the Internet use of children in the Netherlands, we use U.S. statistics as a fundament for our research. In many aspects (technology, social security, welfare, culture) both countries seem rather comparative.

An estimated 65 million children between 2 and 17 have access to the Internet at home in Europe and the US ([34]).

Internet Activity	Ages 8-10	Ages 11-14
School assignments	31.9%	54.3%
Email and instant messaging	26.8%	47.0%
Playing games	33.7%	43.0%
Check news, weather, sports	13.0%	27.4%
Find product information	9.9%	23.6%
Chat rooms	3.5%	14.3%
Watch listen to TV,movies or radio	6.5%	13.0%

(SOURCE: U.S. Census Bureau, Current Population Survey, September 2001.)[32]

Table A.4: Types of activities on internet.

In the Netherlands almost 98% of the children are online, of which 88% use the Internet at home on a daily bases (research by Qirius 2004) [50]. Based on data collected in the United States, 90% of kids 8-12 make use of a computer. At the age of 8, almost 40% of the children make use of internet, growing steadily with every year of age to about 65% for kids of 12 years ([32]).

An important consideration in the design process is keeping in mind where (at school or at home), how, and why (for educational and entertainment needs) kids will use such a portal, this must therefor be investigated during the requirements analysis phase. Furthermore, the importance of the users task when seeking information determines how the user both seeks and processes the information (Cole, 2000; Spink & Cole, 2001) [75].

In order to make a tool to aid children to traverse the Internet, we must thus know for which purposes the user group uses or would like to use the Internet for. The following table gives an indication of the purposes for which children in our age group use the Internet (note: the percentages of children doing activities are given):

Girls are very different types of media and technology users than boys, with communication being the primary benefit, while for the latter research, games, and more solitary activities are the primary usage ([37]). Boys spend more time alone with computers, while girls spent more time using computers with a parent. However, despite the differences in preferences, their tasks can roughly be subdivided into the same categories and most conclusions regarding good Web design for kids hold equally true for boys and girls ([69]).

For optimal usage and applicability, attracting users for a broad spectrum of tasks, we strive to create a portal that achieves a combination of all three of the following objectives:

- entertainment: objective is leisure and fun.
- education: places objective facts and events inside a learning context that has an objective structure and is determined outside the user (i.e. homework assignment given by a teacher).
- information: comprises objective facts and events that are placed within the user's subjective knowledge structure for some kind of intended use.

Search of Information

The question we posed ourselves in section 3.2.1 is "What is search of information?". In order to be capable of answering this question we must understand what information is, and what it means to search this information.

Information

Information is defined as factual data that is capable of increasing one's knowledge or understanding. There is discrepancy between data and information. A bunch of characters or words on a Web page that don't make any apparent sense to the reader represent 'data', data on its own has no meaning but becomes information when it is interpreted. In order to achieve any form of interpretation the child must be capable of reception (i.e. it must make sense, able to read and understand it) and must adhere to his or her interests ([97],[94]), otherwise it will merely remain data.

The following sections describe each of these aspects in detail. The first explains what type of things a child is capable of reading and understanding. The second investigates what type of topics are of interest to our user group.

Reading and Understanding

That a child is capable of reading and understanding a text generally depends on two things, the reading level of the child compared to that required to read the text and level or complexity of its content.

Reading Level

Our research has indicated that children can only determine data to be information when they are appropriately capable of reading and understanding it. Several reading formulas exist that are capable of giving an indication of reading difficulty of a text.

- The Fry Readability Graph (FRG) has been validated for Spanish and English-language documents. The FRG uses three sample passages of text, each exactly 100 words in length, from the beginning, middle, and end of the source document. The grade level is computed as a function of the number of sentences and words contained in the three samples of text. Application of the FRG to Spanish-language documents is similar to its application to English-language documents, with the exception of syllable counting. In Spanish an adjustment compensates for the fact that Spanish text contains more syllables per word than English text of the same reading level. ([95])
- The SMOG (only applicable to English-language documents) uses three passages of 10 sentences each from the beginning, middle and end of the source. If the document has fewer than 30 sentences, some rules and a conversion table are used to calculate the grade level. The reading level is a function only of the number of polysyllabic words (words with three or more syllables) in the sampled text, with more polysyllabic words corresponding to higher reading levels. ([99])
- The Lexile Framework is a relatively new software program that estimates the readability level of a document based on two factors: average sentence length and word familiarity. Passages consisting of shorter sentences are assumed to be easier to read than passages consisting of longer sentences. Passages consisting of familiar (commonly used) words are assumed to be easier to read than passages consisting of unfamiliar words ([100]). Word familiarity is measured by the frequency with

Grade	Lexile
4	650L to 850L
5	750L to 950L
6	850L to 1050L
7	950L to 1075L

Table A.5: The Lexile Framework ([103]).

which a given word is used in written United States school texts of various grade levels ([96]). In this study, the Lexile Framework software was applied to three 10-sentence sample passages drawn from the beginning, middle and end of the source documents. The Lexile scale is a developmental scale for reading ranging from 200L for beginning readers to above 1700L for advanced text.

Though Lexile measures do not translate specifically to grade levels because, within any specific grade, there will be some readers that are far ahead of the rest, and some readers far below the rest. However, numerous studies with large samples of students have observed the approximate reading levels described in table A.5.

Lexile measures can be used to manage comprehension. Matching a readers Lexile measure to a text with the same Lexile measure leads to an expected 75% comprehension rate not too difficult to be frustrating, but difficult enough to be challenging and to encourage reading progress. Advised is to choose texts which span 50L above and 100L below their Lexile measure. Thus, for the entire user group we would choose for texts ranging from 600L to 1175L.

Content Level

Furthermore of interest is that the topics associate to their previous knowledge. Any person involved with didactics or pedagogy will tell you that in order for someone to learn something, one must reflect on and associate with a previous knowledge, because knowledge is built up incrementally rather than by break-through.

A lot of research has been done on content levels that associate to a particular age group ([126]). A good example are the choices of books that children are to read in each year at school . Because a complete listing of topics is merely too lengthy to jot down here, a suggestion is given to attain such topics automatically. Any concepts found in the books that are suggested to our age group can be assumed to be understandable concepts. A document prominently containing similar concepts can be assumed to make sense to the child.

Search Topics

In order to determine which topics children find interesting we analyzed the subjects that children search on through the children's internet search engine Yahooligans! We choose for this search engine because it gives a rather complete indication of children's searches, including:

– An up to date list (updated every weekday) of inputs.

– The percentage of users searching for that subject on that given day.

Subject	Number of days in top 50
Games	249
Cartoon Network	249
Music	249
Dogs	249
Disney	104

Table A.6: 2003-2004 Yahooligans! top searched subjects by children ([84]).

Subject	Number of days in top 50
Dictionary	45
Maps	30
Plants	24
Hurricanes	19
History	1
Science Fair Projects	1

Table A.7: 2003-2004 Yahooligans! top informative searched subjects by children ([84]).

- How long a term has been in the top 50 search terms since September 6, 2003, resulting in a less hype-dependent list.
- Filtering of inappropriate search terms resulting in a list of subjects that are interesting to the broadest possible audience of children.

Some aspects catch the eye. The search topics that have been in the top 50 score the longest are also those that are searched by the most amount of users on a particular day. Because we are not so interested in hypes that relate to things happening in the world in a particular day, we list the most popular all time searches in table A.6.

The highest scoring and longest standing top searched subjects are for leisure purposes, indicating that children more frequently use search engines to search things out of own curiosity than to find the answer to a specific homework question.

The top informative search subjects are listed in table A.7. The reason we researched the most common topics is to understand what children consider interesting. This gives an answer to ”what” children search on internet, and thus also what they consider to be information rather than just data.

Search

Search is defined as an active pursuit or effort to finding or discovering something. This is to say, the user must be proactively exploring possible places with the goal to find information. What types of data are considered information has been discussed previously in this section.

We can distinguish between three different types of expected search results:

1. a short fact (a number or date) as the specific answer to a question;
2. a list of links (such as a directory of games or informative links) as an intermediate result yielding overview and increasing choice and possibilities;

3. a general document including the answer but also giving much more related information about the same topic, often used for reports or presentations.

A.4.7 Results Investigation Children

The following summarizes the answers to the research questions posed in section A.4.1. For a complete view we refer to the appropriate sections above.

Support and ease of use:
A tool which is similar to existing tools that children work with is simple and rather intuitive for a child to work with. Most children in our focus group are acquainted and feel comfortable with the use of standard applications. Our product must focus on using existing and recognizable functionalities.

Children don't know how to appropriately refine their search; the determination of additional keywords poses a large barrier to them. However, they are capable of selecting appropriate keywords from a list of relevant and irrelevant terms. Offering predetermined links lists to aid the selection of sources simplify search tasks (this includes a list of games (or game directories), a dictionary (or thesaurus), an encyclopedia, an atlas). Any interface, especially one designed to help or give assistance, must be specifically designed and written for children, or they won't read or use it.

Children are willing read instructions as long as they are short sentences and the page seems attractive enough to both catch their eye and show that it has been made for children. They are however, extremely impatient, expect fast results and make decisions about relevance within split seconds. This means elements of attractiveness and colors are very important for catching attention, short instructions for giving support and assistance, as well as speed and performance for usability.

Microsoft Windows is the only operating system used by our focus group. All users are acquainted with Microsoft Office products (and its functionalities and toolbars) such as Word.

Their favorite site is often not much more than a list of links to other entertaining sites. In an effort to making our product their starting page, we should offer a location where users can place their own links to their favorite pages.

Children are not capable of performing nor understanding Boolean searches and its results. Furthermore, advanced search and help functionalities are not consulted due to their complexity and dull appearance. Specific options which must be memorized and can be input in the query field (such as 'define:' are not used). These functionalities, if offered, should be offered on a separate page. In any case, they should be adapted to offer extra assistance to our youthful users.

Search for information:
For specific fact seeking, the most popular starting site is Goggle, a query based search engine. For information gathering, this strategy is interleaved with hyperlink browsing. While browsing for entertainment, most children favor a directory listing of games and jokes, such as www.spele.nl. Their most prominent search objectives are information gathering with less well-defined information objectives, often in search of more information about a particular topic, the second is browsing out of curiosity. In both cases, the information need formulation is constrained to a single term, children seem incapable of expansion at this point, due to a lack of domain knowledge.

In order to be considered information, data must be presented in an attractive manner, have pictures, coincide with reading levels, the topic must be interesting to a child, and not pertain to blacklisted topics (see section 3.4.2). Expected results are complete documents, from which sections are copy-pasted into other documents and the pictures are printed.

Their search is both driven by education and entertainment. In both cases children prefer attractive sites with lots of pictures. Dull websites with little colors are often closed almost immediately. Lengthy stories are also not found interesting. However, very short documents are often also not found interesting because they only give a limited amount of information and do not answer the general curiosity of the searcher.

Their search goal is generally information gathering less well-defined information objective. Search of specific facts is rare. Browsing out of curiosity does occur, but usually for entertainment purposes in pursuit of games or quizzes. Children generally expect the result of a search to be a document giving more information pertaining to a topic than they may have explicitly asked for. Preliminary to the search they are often not completely aware of the scale of their knowledge gap and are very satisfied when surplus information is given pertaining to the same topic. As finalization of a search, information is often copy-pasted to a new text document which is then saved. Sometimes the results are printed, but this often only occurs for the pictures; the resulting text is not consulted after printing.

Children show running into troubles during the formulation of their information need. They have a particular expectation, but are incapable of translating that into words. Furthermore, the adjustment of the query poses a big barrier. Children may be capable of coming up with particular terms, but cannot think of synonyms or variants on spelling (other morphological forms of terms) which makes them very frustrated.

Based on the investigation pertaining to the usability of existing tools, the following concrete results have been obtained:

Existing Tool Characteristics to Keep

- **Keyword Query Search:** This functionality is standard, yields quick results, is intuitive and its extreme popularity indicate absolutely no reason to leave it out.

- **Directory:** The use of directories does not burden the child with Boolean syntax because children do not need to distinguish between an intersection search query and a request for a union search. This lightens the cognitive complexity of the task immensely, allowing children to first focus solely on identifying the proper parameters to conduct the search they have in mind. However, in order to be functional and efficient, the number of categories should be kept to a minimum.

- **Relationship-map:** This functionality is intuitive, yet not known to many users. It can dramatically help narrow down search results and aid in the formulation of information need. Furthermore, motivation for including a directory also apply to the relationship-map.

- **Undo/Redo:** The functionality to undo and redo moves has shown to be very helpful and is often used. It allows the user to quickly recover from incorrect decisions, steps chosen, and search strategies.

- **Metaphor:** An animated character is attractive and livens up the interface. It furthermore makes children less restless and can distract them while the system is calculating results.

Existing Tool Characteristics to Adjust

- **Similar Pages:** This has shown to be a very useful functionality. However, the capability is shown before the child has had the opportunity to view the page. Upon viewing the page, or after indicating that a result is relevant, the capability should be shown and emphasized.
- **Advanced Search:** Most children are not capable of adequately using boolean searches not understand its results. However, for the novice users we include the functionality. Contrary to existing tools with such a functionality, a better explanation should be given about its meaning and effect on results. An example should be included.
- **Help functions:** Three types of help functionalities should be included: (1) An online help indicating possible functionalities and including a guided tour with a search example to show how and when these functionalities are used, (2) and context-sensitive help function composed of very short explanations that pop-up when the mouse hovers over a functionality (extremely useful when the user is mine-sweeping the page in search of site capabilities), and (3) a tip-of-the-day which gives a short explanation of a functionality on the page (it should be possible to turn this functionality off to avoid user irritation, and tips are to be chosen at random to avoid boring the user).
- **Spell check:** Spell checking and offering alternatives to query input is useful for children because they often make typos or don't really know how to spell words (especially if they don't know much about it!).
- **Results list:** Because children do not look at all the results in a results list before query reformulation, the results list should be shortened to not burden the user. A maximum of six results should be shown.

Existing Tool Characteristics to Disregard

- **Calculator:** Though it seems like a useful functionality, children have indicated not to do math behind the computer. Furthermore, every computer is equipped with a standard calculator making this functionality superfluous.

New Characteristics to Implement

- **Informative Links list:** A list of informative links aids children in selecting reliable sources of information.
- **Personalized links list:** A location for users to compose their own favorites list of links will increase the chance of our product becoming their starting page, and our research has shown that this dramatically increases the chance of them using this particular page to initiate their search process.

- **Save intermediate results:** Because children are limited in their time and concentration span behind the computer, they often break off their search prior to finalization. Saving and loading intermediate results should be offered.
- **Personalize site:** The capability to personalize the site as changing background or toggling features increases attractiveness of the site and binds the user, increasing the chance that they will return on subsequent visits, as does the function to make our site the browser's starting site.
- **Keyword alternatives:** Offering alternative synonyms and related words to the input query can very well help a child to make an appropriate selection as they are themselves not capable of describing their information gap. Stemming keywords in documents and input query could also dramatically improve results.
- **User relevance feedback:** Children often reformulate queries over and over again without improved results. Using user relevance feedback has proven to yield great results. A manner to force (with burdening or aggravating) the user to indicate relevance and using this information to further determine possible relevant results could be very useful.

A.4.8 Conclusion Investigation Children

This section described the investigation of children. Children are not always capable of adequately self-reflecting, especially when we speak of specific details of task steps, effort and time spent, this complicated the research in several ways, but by observations we were able to make adequate conclusions. However, contrary to adults, children are very honest and upright, if they don't like something or feel frustrated, they won't hide it!

We have found that many functionalities of existing tools are interesting to children and they are very capable of working with them. We have also found several functionalities that children are completely incapable of using, and upon trying to do so, become frustrated and aggravated. What we have not found is a site that combines the best practices and leaves out things frustrating to our focus group. The previous section indicates a list of features that such a site (and thus hopefully our product) will comprise of.

Completeness of Information

In our opinion we were able to obtain a rather complete view and all required appropriate information required to answer the research questions. All questions were answered, each by several children, increasing confidence in the answers obtained. The children were very willing and motivating to cooperate, which resulted in elaborate answers to the questions posed.

Quality of Information

For this investigation we used a broad spectrum of children of different ages and backgrounds. However children are not always capable of correct self-reflection of their tasks and the time and effort they spend. An example is a child who said never to use a directory, however, after asking what his favorite games page was he showed me a directory that he said he used on a daily basis. In spite of this, we feel that combining our interviews with observations and literary research, and furthermore combining the information from this chapter with that of other investigations in this requirements analysis (e.g. interviews with carers and educators) will lead to a reliable view. The group of children interviewed came from a range of different backgrounds, countries, towns and ethnical groups. Because of such a broad spectrum of sources we conclude that the quality of the answers to the research questions are high. Obviously, having completed the study with every child in our focus group and by a larger amount of researchers would have yet increased the quality of the results, yet such a research is unattainable.

Rather puzzling were the contradicting results pertaining to the number of concepts used by children during search. While testing the use-case we concluded that children did not enter full phrases because it cost them too much time to type and was furthermore typing-error prone. During the testing of existing tools, the opposite became evident; almost all children typed in full-phrases. The reason for this was probably because during use-case testing the children were told what they had to look for, while during the latter, they read what they had to look for and were able to copy-paste the question. Our investigations have shown that children are generally not given a specific question to answer (as was using in the existing tools investigation) but rather have a topic in mind of which they must find an answer to. All information gathered and results were useful, however, we find the information pertaining to input queries of the use-cases a better reflection of reality than that of the existing tools.

Limitations of Study

The limitations to the specific investigations pertaining to the Use Case verification and the existing tools has previously been discussed in those sections.

Appendix B

Design

B.1 Results Usability Testing

B.1.1 Cognitive Walkthrough

Table B.1 describes the results of the cognitive walkthrough evaluation performed during the design phase (see section 4.7.2).

B.1.2 Heuristic Evaluation

Tables B.2 and B.3 describe the results of the heuristic evaluation performed during the design phase (see section 4.7.3).

B.2 System Walk Through and Justification

This section describes aspects pertaining to the functioning of the system in natural language. All information in this section has been translated into the ORM and UML models in section 4.4.

B.2.1 Document Preprocessing

1. The spider crawls the web in search of documents.

2. The following dictionaries are read by the system: English common words, Spanish common words, Dutch common words, English synonyms, English stopwords (in the future Dutch and Spanish synonyms and stopwords are also to be included).

3. The spider reads the input file and determines if the file must be saved or discarded (depending on URL/blacklist, if no/recent/outdated version file is currently in the database)

4. The spider then determines the files title, tags the document with a unique id for recognition by our system, and removes HTML tags (discarding or saving contents where appropriate).

5. For each word in the input file, the Lexicon:

Impact	High
Window	Main
Usability Issue	Purpose of software not explained
Recommendation	Expand title to include a hint towards functionality
Impact	High
Window	Main
Usability Issue	(History) Selection in directory or relationship-map not indicated anywhere
Recommendation	Show last few steps in search path.
Impact	Moderate
Window	Main
Usability Issue	There are too many search strategies possible, making it difficult to choose where to start.
Recommendation	Increase overview, give explanations
Impact	Moderate
Window	Login
Usability Issue	If user forgets password, user cannot input a new password.
Recommendation	Allow user to input new password after correctly answering an authorization question
Impact	Moderate
Window	Main
Usability Issue	The mouse clock clutters the screen, hard to read.
Recommendation	Replace mouse clock with other cursor
Impact	Low
Window	Login
Usability Issue	Password input is very complex to input and remember.
Recommendation	Remove high restrictions on password (require only minimal of 6 characters)
Impact	Low
Window	Main
Usability Issue	The buttons are not consistent giving a messy appearance.
Recommendation	Use similar buttons on each screen

Table B.1: Cognitive walkthrough findings.

Page	All
Issue	Consistency and standards
Problem	Buttons not uniform
Recommendation	Use consistent buttons
Page	Login
Issue	Consistency and standards
Problem	Font not legible
Recommendation	Use larger font size
Page	All
Issue	Consistency and standards
Problem	Fonts not consistent
Recommendation	Use same font everywhere
Page	Login
Issue	Error prevention
Problem	Fields not checked on input
Recommendation	Check fields for incorrect input Give feedback to user
Page	All
Issue	Visibility of system status
Problem	No indication system is busy during calculations
Recommendation	Make metaphor jump around to distract and show that system has not crashed
Page	Main
Issue	Visibility of system status
Problem	New reweighted query not shown to user
Recommendation	??
Page	Main
Issue	Match between system and the real world
Problem	Categories not clear
Recommendation	Add icon to each category choice
Page	Results page
Issue	User control and freedom
Problem	User does not see escape option
Recommendation	Make clear must select trash or OK button

Table B.2: Results heuristic evaluation. (1/2)

Page	All
Issue	Recognition rather than recall
Problem	No help
Recommendation	Help pages incl. user manual Tip of the day Tip on hover over option
Page	All
Issue	Help users recognize, diagnose, and recover from errors
Problem	No error descriptions
Recommendation	Adequate error descriptions Search tips
Page	All
Issue	Help and documentation
Problem	No help whatsoever
Recommendation	Help pages User manual Search tips

Table B.3: Results heuristic evaluation. (2/2)

- Ignores spaces and punctuation;
- Transforms letters to lower case, removes accents and digits;
- Ignores numbers altogether.
- Determines which language the document is written in, because only Dutch, English and Spanish documents are of interest to us, all other documents are discarded;
- Determines the reading level of the document according to two different well-known algorithms (FRY and SMOG). For each word encountered, its complexity is determined from a syllable dictionary. Only those documents below and slightly above the reading level of our target group are stored, all others are discarded;
- Strips stopwords from the input document by looking them up in a dictionary of stopwords specific to that language.
- Transforms verb forms to their infinitives using Wordnet.
- Applies stemming to the words using the Porter algorithm.

6. The Analyzer then calculates a document vector which describes the document content by representing terms and term-pairs by their frequencies. The remaining word forms are stored in a vector, once alone, and once in combination with its predecessor (to create noun groups). This is later used for comparison to query vectors to determine relevance.

7. The Analyzer determines which prominent concepts exist within the document.

8. The police then determines if the content of the document belongs to a black listing or not, by analyzing words and concepts in the document. If a document is found to

belong to a blacklisting, it is marked as black, its URL base is stored, and all links to which this document points are also considered blacklisting, so their URL bases are also stored in the black-URL list. Its concepts are stored in a black list and the document and its vector are discarded.

9. Otherwise (the document is not blacklisted), the concepts are used to determine to which concept group the document belongs (e.g. a document about cats belongs to the group animals). Concepts in a document that are also popular concepts, are given a bonus. The concepts and their concept groups are stored in the repository alongside the document vector and information from where it came from.

Analyzing Document Content

In order to determine whether or not a query matches a document, merely a vector calculation must take place. However, as the database is filled with extreme amounts of document vectors, and this calculation must me made while the user must wait for answers, efforts to increase efficiency are required. As such, determining which concepts are prominent in a document at moment of preprocessing can help narrow down which documents are to be used for matching with the query. We thus seek a list of concepts (or keywords) which roughly describe what a document is about. In order to be useful, there must be enough concepts to match with the query, yet not too many for efficiency reasons.

Furthermore, we abstract from the specific details of a document slightly. Because concepts are to be meaningful terms, we wish to describe a document that is about 'meat', also by its hierarchical parent term 'food' to compensate for the lack of domain knowledge that children cope with. The importance of the term we introduce will be lower than that of the original concepts, but as such, we can describe a set of similar documents by a general concept.

B.2.2 Directory and Relationship-Map

Word Relations

WordNet is an online lexical reference system which has an extensive database comprised of English nouns, verbs, adjectives and adverbs are organized into synonym sets, each representing one underlying lexical concept [86]. Its structure can be very useful for several of the functionalities to be built in the final product: the directory, the relationship map, and query expansion.

Hypernyms and Hyponyms are each others inverses. They express sub and super categorization, and are generally available for every term (except the top and the bottom nodes). However, each term can have tens of category levels, and the user should not be burdened with each, thus a selection must be made.

Holonyms and Meronyms are each others inverses. They express the 'part of' relationship. Not every term has known holonyms or meronyms. As synonyms are other words with comparitive meaning, synonyms can be used for query expansion.

Directory Construction

After analysis of existing (adult and childrens) directory listings we have comprised a high-level directory structure of our own which both adheres to the cognitive characteristics of children which came to light during the Requirements Analysis phase (a choice of 6 to 8 categories) and has (to children) meaningful categories and names. The following structure was devised:

- World
- Arts and Entertainment
- Computers and Games
- Science and Nature
- Sports and Recreation
- School

The structure has been shown to the children in the target group, and after having made slight adjustments, the version above has been approved.

In order to achieve a clustering of the documents into this structure, the following steps must be followed:

1. Retrieve sample websites from acknowledged childrens website (store all relevant information such as parent URL, URL, date last updated, and obviously content);
2. Determine the language of the sample document, and discard documents in a language not relevant;
3. Remove stopwords to increase efficiency;
4. Normalize spelling variations using the alternative spelling lists retrieved from WordNet;
5. Place documents in a vector;
6. The most frequently occurring terms are determined from the document. In order to avoid losing extreme precision, the top 5% of the terms in the document are taken rather than a static amount.
7. For the most frequently occurring terms in the document, determine to which hypernym category they belong to.
8. Use statistical analysis to determine the occurrence frequency of terms and their WordNet high level categories, possibly creating a model vector describing all vectors belonging to the category;
9. Cluster documents into predefined categories based on the statistical analysis. If a model vector has been made for the category, it can be determined if a vector belongs to a category by merely calculating cosine similarity between category and document vectors.

Expanding the concept terms using hypernyms or given categories to which terms belong, didnt yield the results we expected. The result was that the relevance score between good matches were extremely low, too low to be useful.

B.2.3 Execute Search

Query Input

1. The user types in one or more keywords into the query input field. The active search retrieves the query from interface.

2. The lexicon translates the query into lower case, removing digits, spaces and punctuation.

3. Stopwords are removed from the input query (the language to be used has been set by the user, alternatively the browser language can be used).

4. Terms are checked for alternative spellings (and replaced) using the alternative spellings lists derived from Wordnet (nouns_alternative_spellings, verbs_alternative_spellings, adjectives_alternative_spellings, adverbs_alternative_spellings). Verbs are replaced by their infinitive form by using Wordnet (verb.exc) if in list of irregular verbs. If a syntactical unit (adverb, adjective, noun, verb) can be determined, this is stored.

5. If no syntactical unit has yet been found, the term is searched in the word lists and its corresponding syntactical unit is stored.

6. If the term has not been found in the word lists (noun_word_id, verb_word_id, adverb_word_id, adjective_word_id), a spelling mistake is assumed. Using edit distance, alternative spelling variations are to be yielded and printed to screen where appropriate.

7. Concatenate term groups of two terms into phrases. Repeat the previous step with each of the phrases.

8. For each noun, synonyms are yielded which are used to expand the input query (Additionally, this may also be done for verbs).

9. Stemming is applied to all query terms.

10. The analyzer translates the resulting query into a vector.

11. The analyzer compares the query vector to the document vectors in the database to find relevant documents (containing relevant and popular concepts) and yields a ranked results list.

12. The active search shows the user the results list in a form desirable by the user.

13. The user selects a document to view.

14. Active search retrieves the selected document from the database.

15. The resulting document is shown in a form desirable by the user.

16. The user indicates whether or not the document is relevant (or this information is gathered by viewing the users actions).

17. The relevance information is used by the analyzer to recompute/reweigh the input query. Concepts of relevantly marked documents are stored in the database as popular concepts.

18. The results list is determined again and the previous steps repeat until the user stops searching.

Directory Search

In constructing the directory structure, as described in section B.2.2, the highest level of hypernyms can be taken as the starting point. Upon selection of a category, the hyponyms of the term must be found and the new directory structure shown.

Relationship-Map

In constructing the relationship-map, both the hypernyms (showing the supercategories of the input/selected term), hyponyms (showing the subcategories of the input/selected term) and meronyms (showing constituents of the selected term) are to be used and presented in a map structure.

The number of terms placed in the map must be restricted to ensure the user maintains overview.

B.3 Conceptual Schema and Relational Database Design

B.3.1 What Information Must Be Stored?

This chapter describes which data needs to stored in the database to enable Giggle to have the functionalities described in the Requirements Analysis document. This role this information plays has graphically been represented in figure 4.3.

Documents

- **documents**:
  There is a document with ID `<document_id>`
- **document_type**:
  Document with ID `<document_id>` is of type `<type>`
- **document_url**:
  Document with ID `<document_id>` has URL `<url>`
- **document_updated**:
  Document with ID `<document_id>` is last updated at `<date>`
- **document_fetched**:
  Document with ID `<document_id>` is fetched at `<date>`
- **document_title**:
  Document with ID `<document_id>` has title `<title>`

- **document_content**:
  The content of the document with ID `<document_id>` is `<content>`

- **document_concept**:
  Document with ID `<document_id>` has concept `<concept_id>`

- **document_language**:
  Document with ID `<document_id>` is in language `<language_doc>`

- **document_reading_level**:
  Document with ID `<document_id>` has reading level `<reading_level>`

- **document_score**:
  Document with ID `<document_id>` has a relevance score of `<relevance_score>`

- **document_color**:
  Document with ID `<document_id>` is listed on the `<doc_listing>` list, which are approved for the user

Value restrictions:

- `<doc_listing>` ∈ { W=White list, G=Gray list } (We don't save documents of the blacklist)
  Initial value is "G"

- `<language_doc>` ∈ { EN=English, NL=Dutch, ES=Spanish }
  Initial value is "EN"

- `<score>`: score is number of relevant hits

User Information

- **users**:
  There is a user with login `<user_id>`

- **user_password**:
  User with login `<user_id>` has the encrypted password `<password>`

- **user_authentication_question**:
  If user with login `<user_id>` has forgotten his/her password, question `<authentication_question_id>` will be asked for access

- **user_authentication_answer**:
  The answer to the access question of user with login `<user_id>` is `<authentication_answer>`

- **user_date_last_visit**:
  The last time the user with login `<user_id>` logged in at `<date>`

- **user_metaphor**:
  User with login `<user_id>` wants the metaphor `<on/off>`

- **user_tip_of_the_day**:
  User with login `<user_id>` wants the tip of the day `<on/off>`
- **user_language**:
  User with login `<user_id>` wants the search results and the GUI of Giggle in the language `<language_doc>`
- **user_background_color**:
  User with login `<user_id>` wants the background to be `<bg_color>`
- **user_favorite_site**:
  User with login `<user_id>` wants to have the link `<url>` behind button number `<DEFANGED_link_position`
- **user_label_favorite_site**.
  Label on the button number `<DEFANGED_link_position>` of user with login `<user_id>` must say `<label_url>`

Value restrictions:

- `<bg_color>` ∈ { BL=Blue, GR=Green, RE=Red, YE=Yellow, WH=White }
  Initial value is "BL"
- `<language_doc>` ∈ { EN=English, NL=Dutch, ES=Spanish }
  Initial value is "EN"
- `<on/off>` ∈ { Y=on, N=off }
  Initial value is "Y" for both metaphor and tip of the day
- `<url>`: Only URLs which are not on the blacklist are aloud

Search States

- **search_states**:
  User with login `<user_id>` has a search query in state `<state>`
- **initial_search_query**:
  The search query of the state `<state>` of user with login `<user_id>` is `<query>`
- **reweighed_search_query**:
  The reweighed search query of the state `<state>` of user with login `<user_id>` is `<query>`
- **search_date**
  The search query of the state `<state>` of user with login `<user_id>` is used at `<date>`
- **relevant_marked_docs**:
  Documents with ID `<document_ids>` are marked by the search query of the state `<state>` of user with login `<user_id>` as relevant
- **irrelevant_marked_docs**:
  Documents with ID `<document_ids>` are marked by the search query of the state `<state>` of user with login `<user_id>` as irrelevant

- **not_marked_docs**:
  Documents with ID `<document_ids>` are by the search query of the state `<state>` of user with login `<user_id>` not marked yet

- **search_path**:
  User with login `<user_id>` came by the search query of the state `<state>` by following the path `<search_path>`

Value restrictions:

- `<document_ids>`: is an enumeration of document IDs, which must be known in table "documents", separated by ";"
  Furthermore, a document (ID) of table "documents" can contain at most in one of the `<document_ids>` of the tables "relevant_marked_docs", "irrelevant_marked_docs", or "not_marked_docs" for a particular user/state combination

- `<state>` $\in$ { C=current query, S=saved query }, later this can be extended such that more then one query can be saved

Blacklist

- **blacklist**:
  Document with base-URL `<url_base>` is listed on the blacklist

Concepts

- **concepts**:
  There is a concept with ID `<concept_id>`

- **concept_keyword**:
  Concept with ID `<concept_id>` contains keyword `<keyword>`

- **concept_color**:
  Concept with ID `<concept_id>` is colored `<doc_listing>`

- **concept_score**:
  Concept with ID `<concept_id>` has a relevance score of `<score>`

Value restrictions:

- `<doc_listing>` $\in$ { W=White, G=Gray, B=Black }

Word Lists

- **hypernym**:
  Term `<term1>` is more generic than term `<term2>`

- **meronym**:
  Term `<term1>` is built up of term `<term2>` (for example, "brim" and "crown" are meronyms of "hat"

- **synonym**:
  A synonym of term `<term1>` is term `<term2>`

Other Information

- **authentication_questions**:
  Authentication question with ID `<authentication_question_id>` ask the user `<authentication_question>`
- **help_pages**:
  There is a help page with ID `<help_id>`
- **help_page_content**:
  On help page with ID `<help_id>` there is shown the information `<content>`
- **keyword_documents**:
  Keyword `<keyword>` is contained in documents with ID `<document_ids>`

Value restrictions:

- `<document_ids>`: is an enumeration of document IDs, which must be known in table "documents", separated by ";"
- `<keyword>`: A keyword in a site is stemmed and is not a stop word (such as 'a', 'the')

B.3.2 Resulting Tables

This section gives an overview of the tables, including the SQL code for creating them in the database server DB2, derived from the information described in the previous section. Figure 4.4 gives a graphical overview.

Different column types are used:

- character(x): An array of characters, which can have a length of maximal 254 characters. This type will allocate for each value a width of x characters, and does not look at the real necessary space.
- clob(x): A chunk of characters with a length of maximal x bytes.
- date
- integer: A digit $\in \aleph$, with a maximum value of of 2,147,483,647.
- smallint: A digit $\in \aleph$, with a maximum value of 32,767.
- varchar(x): An array of characters which can vary in length with a maximum of 'x'. Maximum value of 'x' is 32,672.

Authentication_Questions:

Column name	Type	Optional	Other restrictions
question_id	smallint	no	must be unique (primary key)
question	varchar(254)	no	-

Source code for DB2:

```
CREATE TABLE GIGGLE.AUTHENTICATION_QUESTIONS
      ( QUESTION_ID SMALLINT  NOT NULL ,
          QUESTION VARCHAR(254) NOT NULL ,
          CONSTRAINT PRIMARY_KEY PRIMARY KEY (QUESTION_ID)
        ) IN USERSPACE1 ;
```

Blacklist:

Column name	Type	Optional	Other restrictions
base_url	varchar(254)	no	must be unique (primary key); only the base of the url must be stored

Source code for DB2:

```
CREATE TABLE GIGGLE.BLACKLIST
      ( BASE_URL VARCHAR (254) NOT NULL ,
        CONSTRAINT PRIMARY_BLACKLIST PRIMARY KEY (BASE_URL)
      ) IN USERSPACE1 ;
```

Concepts:

Column name	Type	Optional	Other restrictions
concept_id	integer	no	must be unique (primary key)
doc_listing	character(1)	no	value must be in { B=Black, G=Gray, W=White }; default value is ”G”
relevance_score	double	no	default value is 0

Source code for DB2:

```
CREATE TABLE GIGGLE.CONCEPTS
      ( CONCEPT_ID INTEGER NOT NULL ,
        DOC_LISTING CHARACTER (1) NOT NULL WITH DEFAULT 'G' ,
        RELEVANCE_SCORE DOUBLE NOT NULL WITH DEFAULT 0 ,
        CONSTRAINT PRIMARY_CONCEPT PRIMARY KEY (CONCEPT_ID) ,
        CONSTRAINT CHECK_DOC_COLOR CHECK (DOC_LISTING in ('B', 'G', 'W'))
            ENFORCED ENABLE QUERY OPTIMIZATION
      ) IN USERSPACE1 ;
```

Concept_Keyword:

Column name	Type	Optional	Other restrictions
concept_id	integer	no	must be known in table ”concepts”
keyword	varchar(200)	no	-

The combination of ”concept_id” and ”keyword” must be unique.

Source code for DB2:

```
CREATE TABLE GIGGLE.CONCEPT_KEYWORD
      ( CONCEPT_ID INTEGER NOT NULL ,
        KEYWORD VARCHAR (200) NOT NULL ,
        CONSTRAINT PRIMARY_KEY PRIMARY KEY (CONCEPT_ID, KEYWORD) ,
        CONSTRAINT FOREIGN_KEY FOREIGN KEY (CONCEPT_ID)
            REFERENCES GIGGLE.CONCEPTS (CONCEPT_ID)
            ON DELETE CASCADE ON UPDATE NO ACTION ENFORCED
            ENABLE QUERY OPTIMIZATION
        ) IN USERSPACE1 ;
```

Documents:

Column name	Type	Optional	Other restrictions
document_id	integer	no	must be unique (primary key)
type	varchar(6)	no	-
url	varchar(254)	no	-
date_fetched	date	no	-
date_updated	date	no	-
title	varchar(254)	yes	-
content	clob(500 K)	no	-
language_doc	character(2)	no	must be in { EN=English, NL=Dutch, ES=Spanish }; default value is "EN"
doc_listing	character(1)	no	value must be in { B=Black, G=Gray, W=White }; default value is "G"
relevance_score	integer	no	default value is 0

source code for DB2:

```
CREATE TABLE GIGGLE.DOCUMENTS
      ( DOCUMENT_ID INTEGER NOT NULL ,
        TYPE VARCHAR(6) NOT NULL ,
        URL VARCHAR (254) NOT NULL ,
        DATE_FETCHED DATE NOT NULL ,
        DATE_UPDATED DATE NOT NULL ,
        TITLE VARCHAR (254) ,
        CONTENT CLOB (500 K) NOT NULL NOT LOGGED NOT COMPACT ,
        LANGUAGE_DOC VARCHAR (7) NOT NULL WITH DEFAULT 'EN' ,
        DOC_LISTING CHARACTER (2) NOT NULL WITH DEFAULT 'G' ,
        RELEVANCE_SCORE INTEGER NOT NULL WITH DEFAULT 0 ,
        CONSTRAINT PRIMARY_KEY PRIMARY KEY (DOCUMENT_ID) ,
        CONSTRAINT VALUE_LANGUAGE CHECK (LANGUAGE_DOC IN ('EN', 'NL', 'ES'))
            ENFORCED ENABLE QUERY OPTIMIZATION ,
        CONSTRAINT CHECK_DOC_COLOR CHECK (DOC_LISTING in ('B', 'G', 'W'))
            ENFORCED ENABLE QUERY OPTIMIZATION
      ) IN USERSPACE1;
```

The document_id is of type integer. This can be changed to bigint in case more than 4 milliard (integer has a range of -2,147,483,648 to 2,147,483,647) documents must be stored. Note, the column type of document_id in the tables document_concept, and document_reading_level need also be changed.

Document_Concept:

Column name	Type	Optional	Other restrictions
document_id	integer	no	must be known in table ”document”
concept_id	integer	no	must be known in table ”concept”

The combination of ”document_id” and ”concept_id” must be unique.

source code for DB2:

```
CREATE TABLE GIGGLE.DOCUMENT_CONCEPT
      ( DOCUMENT_ID INTEGER NOT NULL ,
        CONCEPT_ID INTEGER NOT NULL  ,
        CONSTRAINT PRIMARY_KEY PRIMARY KEY (DOCUMENT_ID, CONCEPT_ID) ,
        CONSTRAINT FOREIGN_DOC FOREIGN KEY (DOCUMENT_ID)
            REFERENCES GIGGLE.DOCUMENTS (DOCUMENT_ID)
            ON DELETE CASCADE ON UPDATE NO ACTION
            ENFORCED ENABLE QUERY OPTIMIZATION ,
        CONSTRAINT FOREIGN_CONCEPT FOREIGN KEY (CONCEPT_ID)
            REFERENCES GIGGLE.CONCEPTS (CONCEPT_ID)
            ON DELETE CASCADE ON UPDATE NO ACTION
            ENFORCED ENABLE QUERY OPTIMIZATION
      ) IN USERSPACE1 ;
```

Document_Reading_Level:

Column name	Type	Optional	Other restrictions
document_id	integer	no	must be known in table ”document”
reading_level	smallint	no	-

The combination of ”document_id” and ”reading_level” must be unique.

source code for DB2:

```
CREATE TABLE GIGGLE.DOCUMENT_READING_LEVEL
      ( DOCUMENT_ID INTEGER NOT NULL ,
        READING_LEVEL SMALLINT NOT NULL  ,
        CONSTRAINT PRIMARY_KEY PRIMARY KEY (DOCUMENT_ID, READING_LEVEL) ,
        CONSTRAINT FOREIGN_SITE FOREIGN KEY (DOCUMENT_ID)
            REFERENCES GIGGLE.DOCUMENTS (DOCUMENT_ID)
            ON DELETE CASCADE ON UPDATE NO ACTION
            ENFORCED ENABLE QUERY OPTIMIZATION
      ) IN USERSPACE1 ;
```

Help_Pages:

Column name	Type	Optional	Other restrictions
help_id	smallint	no	must be unique (primary key)
content	clob(500 K)	no	-

Source code for DB2:

```
CREATE TABLE GIGGLE.HELP_PAGES
      ( HELP_ID SMALLINT NOT NULL ,
        CONTENT CLOB (500 K) NOT NULL NOT LOGGED NOT COMPACT ,
        CONSTRAINT PRIMARY_KEY PRIMARY KEY (HELP_ID)
      ) IN USERSPACE1 ;
```

Hypernym:

Column name	Type	Optional	Other restrictions
general_term	varchar(200)	no	-
specific_term	varchar(200)	no	-

The combination of "general_term" and "specific_term" must be unique.

Source code for DB2:

```
CREATE TABLE GIGGLE.HYPERNYM
      ( GENERAL_TERM VARCHAR (200) NOT NULL ,
        SPECIFIC_TERM VARCHAR (200) NOT NULL ,
        CONSTRAINT PRIMARY_KEY PRIMARY KEY (GENERAL_TERM, SPECIFIC_TERM)
      ) IN USERSPACE1 ;
```

Keyword_Documents:

Column name	Type	Optional	Other restrictions
keyword	varchar(200)	no	must be unique (primary key); contains only single terms and two terms
document_ids	clob(2147483647)	no	contains a list of document_ids, known in table "documents", separated by ';'

Source code for DB2:

```
CREATE TABLE GIGGLE.KEYWORD_DOCUMENTS
      ( KEYWORD VARCHAR (200) NOT NULL ,
        DOCUMENT_IDS CLOB(2147483647) NOT NULL ,
        CONSTRAINT PRIMARY_KEY PRIMARY KEY (KEYWORD) ,
      ) IN USERSPACE1 ;
```

Meronym:

Column name	Type	Optional	Other restrictions
kind	varchar(200)	no	-
class	varchar(200)	no	-

The combination of "kind" and "class" must be unique.

Source code for DB2:

```
CREATE TABLE GIGGLE.MERONYM
      ( KIND VARCHAR (200) NOT NULL ,
        CLASS VARCHAR (200) NOT NULL ,
        CONSTRAINT PRIMARY_KEY PRIMARY KEY (KIND, CLASS)
      ) IN USERSPACE1 ;
```

Search_States:

Column name	Type	Optional	Other restrictions
user_id	varchar(30)	no	must be known in table "users"
state	character(1)	no	value must be in { C=current query, S=saved query }; default value is "C"
search_date	date	no	default value is current date
initial_search_query	varchar(254)	no	-
reweighed_search_query	varchar(254)	yes	-
relevant_marked_docs	varchar(500)	yes	-
irrelevant_marked_docs	varchar(500)	yes	-
not_marked_docs	varchar(32672)	yes	-
search_path	varchar(10000)	yes	-

The combination of "user_id" and "state" must be unique, and is the primary key of this table. Furthermore, all documents which are marked relevant cannot be marked irrelevant or not been marked. And all documents marked irrelevant cannot be relevant or not marked.

Source code for DB2:

```
CREATE TABLE GIGGLE.SEARCH_STATES
      ( USER_ID VARCHAR (30) NOT NULL ,
        STATE CHARACTER (1) NOT NULL WITH DEFAULT 'C' ,
        SEARCH_DATE DATE NOT NULL WITH DEFAULT CURRENT DATE ,
        INITIAL_SEARCH_QUERY VARCHAR (254) NOT NULL ,
        REWEIGHED_SEARCH_QUERY VARCHAR (254) ,
        RELEVANT_MARKED_DOCS VARCHAR (500) ,
        IRRELEVANT_MARKED_DOCS VARCHAR (500) ,
        NOT_MARKED_DOCS VARCHAR (32672) ,
        SEARCH_PATH VARCHAR (10000) ,
        CONSTRAINT PRIMARY_KEY PRIMARY KEY (USER_ID, STATE) ,
        CONSTRAINT FOREIGN_LOGIN FOREIGN KEY (USER_ID)
            REFERENCES GIGGLE.USERS (USER_ID)
            ON DELETE CASCADE ON UPDATE NO ACTION
            ENFORCED ENABLE QUERY OPTIMIZATION ,
        CONSTRAINT VALUE_RELEVANCE CHECK (STATE IN ('C', 'S'))
            ENFORCED ENABLE QUERY OPTIMIZATION
        ) IN USERSPACE1 ;
```

Synonym:

Column name	Type	Optional	Other restrictions
term	varchar(200)	no	-
synonym	varchar(200)	no	-

The combination of "term" and "synonym" must be unique.

Source code for DB2:

```
CREATE TABLE GIGGLE."SYNONYM"
      ( TERM VARCHAR (80) NOT NULL ,
        SYNONYM_TERM VARCHAR (80) NOT NULL ,
        CONSTRAINT PRIMARY_KEY PRIMARY KEY (TERM, SYNONYM_TERM)
      ) IN USERSPACE1 ;
```

Users:

Column name	Type	Optional	Other restrictions
user_id	varchar(30)	no	must be unique (primary key)
password	varchar(10)	no	must minimal consist of 8 characters of which 2 non-letters
date_last_login	date	no	-
authentication_question_id	smallint	no	must be known in table "authentication_questions"
authentication_answer	varchar(50)	no	-
metaphor	character(1)	no	must be in { Y=on, N=off }; default value is "Y"
tip_of_the_day	character(1)	no	must be in { Y=on, N=off }; default value is "Y"
language_doc	character(2)	no	must be in { EN=English, NL=Dutch, ES=Spanish }; default value is "EN"
bg_color	character(2)	no	must be in table "colors"; default value is "BL" if specified, else the first entry of the table "colors"

For better adaptability of the background color, the possible colors will be store in a table, which contain the RGB values of these colors. So the administrator can adapt the RGB values of existing colors, and add/remove colors.

Source code for DB2:

```
CREATE TABLE GIGGLE.USERS
      ( USER_ID VARCHAR (30) NOT NULL ,
        PASSWORD VARCHAR (10) NOT NULL ,
        DATE_LAST_LOGIN DATE NOT NULL WITH DEFAULT CURRENT DATE,
        AUTHENTICATION_QUESTION_ID SMALLINT NOT NULL ,
        AUTHENTICATION_ANSWER VARCHAR (50) NOT NULL ,
        METAPHOR CHARACTER(1) NOT NULL WITH DEFAULT 'Y' ,
        TIP_OF_THE_DAY CHARACTER (1) NOT NULL WITH DEFAULT 'Y' ,
        LANGUAGE_DOC CHARACTER (2) NOT NULL WITH DEFAULT 'EN' ,
        BG_COLOR CHARACTER (2) NOT NULL WITH DEFAULT 'BL' ,
        CONSTRAINT PRIMARY_KEY PRIMARY KEY (USER_ID) ,
        FOREIGN_QUESTION FOREIGN KEY (AUTHENTICATION_QUESTION_ID)
            REFERENCES GIGGLE.AUTHENTICATION_QUESTIONS (QUESTION_ID)
            ON DELETE CASCADE ON UPDATE NO ACTION
            ENFORCED ENABLE QUERY OPTIMIZATION ,
        FOREIGN_COLOR FOREIGN KEY (BG_COLOR)
```

```
            REFERENCES GIGGLE.COLORS (COLOR_LABEL)
            ON DELETE CASCADE ON UPDATE NO ACTION
            ENFORCED ENABLE QUERY OPTIMIZATION ,
        CONSTRAINT LANGUAGES CHECK (LANGUAGE_DOC IN ('EN', 'NL', 'ES'))
            ENFORCED ENABLE QUERY OPTIMIZATION ,
        CONSTRAINT VALUES_METAPHOR (METAPHOR IN ('Y', 'N'))
            ENFORCED ENABLE QUERY OPTIMIZATION ,
        CONSTRAINT VALUES_TIP_DAY (TIP_OF_THE_DAY IN ('Y', 'N'))
            ENFORCED ENABLE QUERY OPTIMIZATION
      ) IN USERSPACE1 ;
```

Colors:

Column name	Type	Optional	Other restrictions
color_label	character(2)	no	must be unique (primary key)
r_value	smallint	no	must be between 0 and 255
g_value	smallint	no	must be between 0 and 255
b_value	smallint	no	must be between 0 and 255

Initial the colors blue, green, red, white, and yellow will be specified.

Source code for DB2:

```
CREATE TABLE GIGGLE.COLORS
      ( COLOR_LABEL CHARACTER (2) NOT NULL ,
        R_VALUE SMALLINT NOT NULL ,
        G_VALUE SMALLINT NOT NULL ,
        B_VALUE SMALLINT NOT NULL ,
        CONSTRAINT PRIMARY_KEY PRIMARY KEY (COLOR_LABEL) ,
        CONSTRAINT R_VALUE_RANGE CHECK (R_VALUE >= 0 AND R_VALUE <= 255)
            ENFORCED ENABLE QUERY OPTIMIZATION ,
        CONSTRAINT G_VALUE_RANGE CHECK (G_VALUE >= 0 AND G_VALUE <= 255)
            ENFORCED ENABLE QUERY OPTIMIZATION ,
        CONSTRAINT B_VALUE_RANGE CHECK (B_VALUE >= 0 AND B_VALUE <= 255)
            ENFORCED ENABLE QUERY OPTIMIZATION
      ) IN USERSPACE1 ;
```

User_Favorite_Site:

Column name	Type	Optional	Other restrictions
user_id	varchar(30)	no	must be known in table "users"
link_position	smallint	no	-
url	varchar(254)	no	-
label_url	varchar(25)	no	-

The combination of "user_id" and "link_position" must be unique.

Source code for DB2:

```
CREATE TABLE GIGGLE.USER_FAVORITE_SITE
      ( USER_ID VARCHAR (30) NOT NULL ,
    LINK_POSITION SMALLINT NOT NULL ,
    URL VARCHAR (254) ,
    LABEL_URL VARCHAR (25) ,
    CONSTRAINT PRIMARY_KEY PRIMARY KEY (USER_ID, URL) ,
    FOREIGN_LOGIN FOREIGN KEY (USER_ID)
        REFERENCES GIGGLE.USERS (USER_ID)
        ON DELETE CASCADE ON UPDATE NO ACTION
        ENFORCED ENABLE QUERY OPTIMIZATION
  ) IN USERSPACE1 ;
```

Appendix C

Testing

C.1 Test Log

In this section we document all the results of functional testing described in chapter 7. For purposes of enforcing structure and overview and a systematic approach, we make a distinction between unit testing, integration testing, and system testing.

- **Unit Testing:** Results of unit testing are shown in tables C.1 and C.2.
- **Integration Testing:** Results of integration testing are shown in table C.3.
- **System Testing:** Results of system testing are shown in table C.4.

Error	**Action**	**Priority**	**State**
Active Search State			
Analyzer			
Document length incorrect	Correct	High	√
Normalize over collection	Implement	Medium	√
Database			
Errors not caught	Catch errors accordingly	High	√
Yielded errors not clear	Specify defined errors	High	√
Problems input large files	Break files down into portions	High	√
Attempt duplicate input	Check if input already in table	High	√
Long terms not input in tables	Increase column length	High	√
Directory			
Doc-dir determination not suff.	Extend training	High	X
Document Preprocessing			
Duplicate in tf incorrect	increment duplicates	High	√
Help			
No help content	Write content	Medium	√
Interface			
No options hovering labels	Create labels on options	Low	√
No tool description	Add description to title	High	√
No metaphor on screen	Set Giggle on screen	Low	√
Language Determiner			
Only germanic languages	Other alphabets not required	Low	√
No distinc. Spanish/Italian	Expand word lists/gram. rules	Low	X
No distinc. Dutch/German	Expand word lists/gram. rules	Low	X
Lexicon			
Accents not stripped	Strip all accents	High	√
Login			
Police			
Threshold for gray documents	Determine and implement	High	√
Set black URLs	Set in dbase	High	√
Query Input			
Synonyms too important	Adjust weight synonyms	High	√
Spelling alts too important	Adjust weight spelling alts	High	√
Reading-Level Determiner			
Level too high	Normalize reading level	Medium	X

Table C.1: Results of Unit Testing (1/2)

Error	**Action**	**Priority**	**State**
Relationship-Map			
Too many subterms	Determine sub terms	High	X
Multiple term query input	Determine important terms	High	√
Resulting Document			
Search terms not clear	Highlight search terms	High	√
No link to original page	Add link to orig. page	Medium	√
No warning when leave site	Add danger warning	Low	X
Results List			
No summary	Determine summary	High	√
Spell Check			
Yields far too many alternatives	Limit alternatives	High	√
Not capable of Dutch	Implement Dutch capabilities	Low	X
Not capable of Spanish	Implement Spanish capabilities	Low	X
Spider			
Black docs	Check if docs from black URL	High	√
Date retrieved wrong	Fix date retrieved	Medium	X
Stemmer			
Can't transform plural to singular	Code possibility	High	√
Error stemming phrases	Parse stem	High	√
Tagger			
Only HTML codes	Expand for multiple formats	Low	X
User Relevance Feedback			
No user info incorp.	Attain user info	High	√

Table C.2: Results of Unit Testing (2/2)

Error	**Action**	**Priority**	**State**
Retrieval Functionalities			
Rel content not recorded	Set rel in dbase	Medium	√
Document Preparation			
Language test missing	Apply language test	High	√
Read level missing	Apply read level	High	√
Database			
Table description	Update	High	√
User Personalization			
User_id lost w search state	Store user_id-search	High	√
Interface			
Search path history	Clarify clickable	High	√
Directory	Clarify buttons	High	√
Tip of the day	Implement	Low	√

Table C.3: Results of Integration Testing

Error	**Action**	**Priority**	**State**
Database			
Very poor performance	Optimize database	Low	√
Error on spider update	Debug update	Medium	√
Document Preparation			
Filtering Adds and Banner	Filter	Low	X
Filtering Pop-ups	Filter	Low	√
Interface			
Missing Introduction	Include intro	Medium	√
Directory not refreshed	Refresh directory	High	√
Help not available from every page	Make available	High	√
SQL error during login	Handle error	High	√
Spider error during update	Handle error	High	√
Logout not possible	Logout functionality	Low	X
Slow response	Optimize responsiveness	Low	X
Results list empty	Offer alternative action	Low	X

Table C.4: Results of System Testing

Nr.	Description	Feature	P/S/D	Result
F1	Instructions/Help	On-line Tutorial Context Sensitive Help	P P	✓ ✓
F2	Metaphor	Character	P	✓
F3	Initiate Searching Process	Directory Example Search	P P	✓ X
F4	Formulation Information Need	Keyword query input Directory Relationship-map Natural-language query input	P P P D	✓ ✓ ✓ X
F5	Aid Query Reformulation	Query expansion Specification category/rel-map	P P	✓ ✓
F6	Following Progress Search Process	Modified query Path chosen (clickable)	P P	✓ ✓
F7	Relevance Feedback	Store intermediate results Relationship query+result	P P	✓ ✓
F8	Reversal of Actions	Undo/Step back Redo/Step forward	P P	✓ ✓
F9	Navigation and Search Strategies	Keyword query input Directory Relationship map Corresponding components Advanced search Query input tips	P P P P S P	✓ ✓ ✓ ✓ X ✓
F10	Sitemap	Map	D	X

Table C.5: Test results functional requirements.

Nr.	Description	Feature	P/S/D	Result
NF1	Usability	Easy to learn/use	P	✓
		Multi-user	D	X
NF2	Presentation	Interface guidelines	P	✓
		Make Giggle start page	P	✓
		Login and personalization	P	✓
		Block popups, banners, ads	P	X
NF3	Relevance and Filtering	Filter inappropriate content	P	✓
		Filter on reading level	P	✓
		Filtering on other aspects	S	✓
		Promote favorable concepts	P	✓
		System learning from user input	S	✓
NF4	Safety Warning	Educate user about safety	P	✓
		Encryption user information	P	X
NF5	Hard and Software	Windows 2000 and up OS	P	✓
		IE and Mozilla browsers	P	✓
		Existing filters	P	✓
NF6	Reaction Time	Adequate performance	P	✓
		High performance	D	X

Table C.6: Test results of non-functional requirements.

Nr.	Description	Feature	P/S/D	Result
B1	Awareness	IBM	P	✓
		School	P	✓
		Community	S	✓
B2	Multi-linguistic	English	P	✓
		Dutch	S	X
		Spanish	S	X

Table C.7: Test results of business requirements.

Appendix D

Glossary

Application Coded instructions that tell the computer to perform specific activities, such as word processing or money management. Often the terms software, programs, applications, and titles are used interchangeably.

Black listing Documents with unappropriate (unsafe, unfit or harmful) content, as specified in the requirements analysis document.

Bookmark An entry in your browser that lists your favorite Web sites and links you to them without your having to retype the addresses; used variously as a noun and a verb.

Browser Software that allows you to view and explore the World Wide Web.

Browsing Interactive task in which the user moves from place to place on the Internet searching for topics of interest, being more interested in exploring the document collection than in specifically retrieving documents which satisfy a particular information need.

Chat A real-time ”conversation” in which participants communicate in writing via computers and the Internet.

Chat room An online service where people can chat with each other by typing messages realtime.

Clustering The grouping of documents which satisfy a set of common properties. The aim is to assemble together documents which are related among themselves. Clustering can be used, for instance, to expand a user query with new and related index terms.

Collection A group of items, often documents.

Crawler A program that search the web for data, such as documents and images.

DB2 Database 2, a family of relational database products offered by IBM.

Design Secondary phase comprising of a one-to-one translation of the requirements into a realizable tool capable of being built.

Directory A hierarchical categorization of concepts in a domain of knowledge, indexes of Web pages organized by subject.

Document A unit of retrieval. It might be a paragraph, a section, a chapter, a Web page, an article, or a whole book.

Domain name A Web site address, usually followed by .com, .org, or .edu.

Download To transfer data from an online source to your computer.

Edit distance (between two strings) The minimum number of insertions, deletions, and replacements of characters necessary to make two strings equal.

E-mail Electronic mail; written messages sent from one computer to another via the Internet.

Enabling Web-product A software program that aids a user to traverse pages and documents on the Internet.

Favorable document A document that adheres to the specifications described in the requirements document, such as reading levels and content topics. It is a subset of white listing documents, and absolutely not an element of either black or gray listing.

Filtering software Software that sorts information on the Internet and classifies it according to content. Some filtering software allows the user to block certain kinds of information on the Internet.

Functional characteristic A characteristic of the final product which has the function to complete a particular task.

Giggle The final project deliverable to be given to the client.

Gray listing Documents of which the listing category has not yet been determined, but their content could potentially be unappropriate.

Holonym The name of the whole of which the meronym names a part. Y is a holonym of X if X is a part of Y .

Hyperlink An image or portion of text on a Web page that is linked to another Web page (either on the same site or in another Web site). Clicking on the link will bring the user to go to another Web page.

Hypernym The generic term used to designate a whole class of specific instances. Y is a hypernym of X if X is a (kind of) Y.

Hyponym The specific term used to designate a member of a class. X is a hyponym of Y if X is a (kind of) Y .

Implementation Process of building a tool by directly translating a design into a physical piece of software. Testing and verification of the tool in this phase ensure that it adheres to the requirements posed on the product; as such, guaranteeing that the translation from requirements to design to implementation has not failed. This phase does not include embedding and commercializing the product to ensure that it can actually be used.

Index term (or keyword) A significant or descriptive term which can be used to refer to the content of a document. Usually, index terms are nouns or noun groups. In the Web, however, some search engines use all the words in a document as index terms.

Information retrieval Retrieval of information (not data) from a collection of written documents. The retrieved documents aim at satisfying a user information need usually expressed in natural language.

Internet An unstructured, unregulated and international network of computers.

Irrelevant document Any document incapable of answering the information need posed by the user. We propose that all documents belonging to the set of black-listing also belong to this set.

Keyword See index term.

Meronym The name of a constituent part of, the substance of, or a member of something. X is a meronym of Y if X is a part of Y .

Metasearch A search technique common on the World Wide Web where a single point of entry is provided to multiple heterogenous back-end search engines. A meta search system sends a user's query to the back-end search engines, combines the results, and returns a single, unified hit-list to the user.

Multimedia A combination of media types including text, graphics, animation, audio and video.

Non-functional characteristic A characteristic of the final product not made to be functional or do a task, but rather create an environment in which the user is capable of working, such as a particular feature or aspect in the user interface incorporated for leisure and enjoyability.

Non-favorable document Any document that is not a favorable document according to one or more of the specifications in the requirements document.

Online Having access to the Internet, World Wide Web, personal communications, and the like.

OO Object-Oriented, a software engineering method for analysis, design, programming and testing.

Precision An information retrieval performance measure that quantifies the fraction of retrieved documents which are known to be relevant.

Program See **Tool**.

Query The expression of the user information need in the input language provided by the information system. The most common type of input language simply allows the specification of keywords and of a few Boolean connectives.

Query expansion A process of adding new terms to a given user query in an attempt to provide better contextualization (and hopefully retrieve documents which are more useful to the user).

Recall An information retrieval performance measure that quantifies the fraction of known relevant documents which were effectively retrieved.

Relevant document A document capable of answering the information need posed by the user. This may be an element of a favorable document set or a gray-listing.

Requirement A stakeholder desired target, need, expectation, or constraint.

Requirements analysis Preliminary phase in which all separate elements, requirements, and components of the whole product to be built are identified.

Search engine A tool to help people locate information available on the World Wide Web. By typing in keywords (or making selections from given keywords), users can find numerous Web sites that contain the information sought.

Specification A document that states requirements.

Software A series of computer instructions or data that can be stored electronically.

Sound Logically valid and having true premises.

Stakeholder A stakeholder is a person or company who has an interest or share in an undertaking, in some manner involved in the process or final product. They are the source of the requirements.

Stemming A technique for reducing words to their grammatical roots.

Stopwords Words which occur frequently in the text of a document. Examples of stopwords are articles, prepositions, and conjunctions ("the", "and").

Surfing The process of browsing the Web, by linking from site to site. A less purposeful, more entertaining activity in contrast to "searching" the Web.

Synonym Another word for a particular term having the same meaning.

Tag A string which is used to mark the beginning or ending of structural elements in the text.

Tool Coded instructions that tell the computer to perform specific activities. A software program or product built for a particular aim, to solve a particular problem for a particular user. In our case this product is Giggle.

URL Acronym for Uniform Resource Locator; the unique address for each site on the Web.

Use Case Use cases can be used as a way of describing how a user will interact with the application that is being designed, representing a functional requirement by showing what happens, not how it is achieved by the system. Each use case constitutes a complete course of action (one or more tasks) initiated by a user, and it specifies the interaction that takes place between the user and the system. The collected set of use cases specify all the existing ways of using the system.

User information need A natural language declaration of the information need of a user. For instance "find documents which discuss the rules of the Pokemon red game".

Web Internet.

White listing Documents with benign and safe content with respect to our user group. We assume that documents which can be attained through popular children's sites that are manually-censored by experts, to be safe and belong to this set.

WWW - World Wide Web The hyperlinked text and graphic-based part of the Internet.

w3 IBM's intranet, accessible to all IBM employees for the exchange of information.

Bibliography

[1] http://www.business-ethics.com/100best.htm. Business Ethics' Names 100 Best Corporate Citizens, June 22nd, 2004.

[2] Haas, Pamela. http://www3.gsb.columbia.edu/botline/fall02/1121/SEPanel.html. "The Bottom Line", June 22nd, 2004.

[3] http://www.calvert.com/calvertindexprofile.asp?Profile=IBM.
Socially Responsible Investing - Calvert Social Index, June 20, 2004.

[4] http://www.ecsite-uk.net/news/ecsite/ecsite_newsletter_53_winter_2002.pdf

[5] http://www.vault.com/nr/consulting_rankings/consulting_rankings.jsp?consulting2004=1. Top 50 Consulting Firms, June 22, 2004.

IBM

[6] International Business Machines Coorporation. (2002), "2002 Corporate Responsibility Report".

[7] http://www.ibm.com/ibm/ibmgives/. Corporate Community Relations, May 20, 2004.

[8] http://www-306.ibm.com/ibm/easy/eou_ext.nsf/publish/558. October 5th, 2004.

[9] IBM ASM NL Quality Managment System. (2003) *Macro Design Document*.

[10] IBM ASM NL Quality Managment System. (2003) *Software Design Checklist*.

[11] http://www.ibm.com/privacy/details/us/. IBM privacy practices on the web. September 28th, 2004.

[12] http://www-306.ibm.com/software/rational/. IBM Rational Software. October 26th, 2004.

[13] http://ue.torolab.ibm.com/ue/pages/html/!B0/online/principles.htm. October 5th, 2004.

[14] http://www.ictopschool.net/. Stichting Ict op School voor samenwerking en kennisuitwisseling. October 8th, 2004.

[15] http://www-5.ibm.com/nl/maatschappij/projecten_nederland.html. IBM Corporate Community Projecten, July 6, 2004.

[16] International Business Machines Coorporation. (1999), ”Reinventing Education Grant Program”.

[17] http://www.kidsmartearlylearning.org. Kidsmart Early Learning, May 20, 2004.

[18] http://www.mentorplace.org. IBM MentorPlace, May 20, 2004.

[19] http://www.tryscience.org. Tryscience, May 18, 2004.

[20] http://www.kennisland.nl/kennisland/binaries/KLsite/actueel/publicaties. ff contact :-). (2004) ”FFcontact Actieprogramma 2004”.

Methodology

[21] Ambler, Scott. (2002), *Know the user before implementing a system*, *Computing Canada*, Feb 1, 2002; 28, 3; ABI/INFORM Global, p. 13.

[22] Gendt, J. van. and Weeda, R. (2004), *Distillation Methods*.

[23] Nielsen, J. (1995), *Scenarios in discount usability engineering*, J. Carroll (Ed.), *Scenario-based design: Envisioning work and technology in system development*, pp. 151-167, New York, Wiley.

[24] Pressman, R. (2000), *Software Engineering, A Practitioner's Approach*, 5th edition, Berkshire, McGraw-Hill International.

[25] Saunders, M., Lewis, P., and Thornhill, A. (2003), *Research methods for business students*, 3rd edition, Essex, Pearson Education Limited.

[26] Sommerville, Ian. (2001), *Software Engineering*, 6th edition, Addison Wesley.

[27] http://www.controlchaos.com/manage.htm. ExtremeValue : Business Value Driven Software Development, July 6th, 2004.

[28] http://www.iso.org/iso/en/ISOOnline.frontpage. International Organization for Standardization, January 23rd, 2005.

[29] http://stylusinc.com/website/scrum.htm. SCRUM: Saving Projects from Failing, July 6th, 2004.

Children and internet

[30] ”Porn Websites abuse brand names of toys”. Envisional. Financial Times. November 16th, 2000.

[31] Brehm, Millerm, Perlman, Campbell. (2002), *Intimate relationships*, New York.

[32] DeBell, M. and Chapman, C. (2003), *Computer and Internet Use by Children and Adolescents in 2001.*

[33] Dool, P. and Brummelhuis, A. (2003) Report: *Compileren van Kennis* Ict op school.

[34] ETSI TR 102 133: Technical Report. (2003), *Human Factors(HF); Access to ICT by young people: issues and guidelines*, http://docbox.etsi.org/EC_Files/EC_Files/tr_102133v010101p.pdf.

[35] Hirsh, S.G. (1997), *How do children find information on different types of tasks? Childrens use of the Science Library Catalog*, Library Trends, 45.

[36] Hirsh, S.G. (1999), *Children's Relevance Criteria and Information Seeking on Electronic Resources*, Journal of the American Society for Information Science, 50(14), pp. 1265-1283

[37] Knowledge Networks Statistical Research. (2003), *How Children Use Media Technology 2003*, www.sri.knowledgenetworks.com.

[38] International Baccalaureate Organization. *Middle Years Progamme Technology*, October 2000.

[39] Newburger, E. (2001), *Home Computers and the Internet Use in the United States: August 2000*, Current Population Reports.

[40] Revelle, G., Druin, A., Platner, M., Weng, S., Bederson, B., Hourcade, J., and Sherman, L. (2000), *Young Childrens Search Strategies and Construction of Search Queries*, Human-Computer Interaction Lab University of Maryland.

[41] Inan, Hurol. (2004), *Search User Behaviour Patterns.*

[42] Stegers, E. (2001) *Behoefte aan ondersteuning bij gebruik van computers in het onderwijs.* Nipo het marktonderzoekinstituut, rapport September 2001.

[43] http://www.ala.org/ala/washoff/WOissues/civilliberties/cipaweb/
cipa.htm. Children's Internet Protection Act. American Library Association, August 24, 2004.

[44] http://www.cyberpatrol.com/. CyberPatrol, July 6th, 2004.

[45] http://www.cyveillance.com. Cyveillance Study, March 1999.

[46] http://www.wsu.edu/DrUniverse. Ask Dr. Universe, September 30th, 2004.

[47] http://www.idc2004.org. Interaction Design and Children, July 6th, 2004.

[48] http://www.isafe.org/. i-SAFE America. October 14th, 2004.

[49] http://www.getnetwise.org/. GetNetWise, July 6, 2004.

[50] http://www.kennisnet.nl. Kennisnet: De internetorganisatie for het onderwijs. October 7th, 2004.

[51] http://www.kinderconsument.nl. de Kinder Consument, June 22nd, 2004.

[52] http://www.shocknews.nl/nieuws/928.html. "Internetzuilen bij McDonald's", October 22nd, 2002.

[53] http://www.ouders.nl/krowser.htm. Krowser - de kinderbrowser van Ouders Online, June 22nd, 2004.

[54] http://www.protectkids.com/dangers/childaccess.htm. ProtectKids: Protecting Children in Cyberspace, July 6th, 2004.

[55] http://www.surfopsafe.nl. Surf op Safe, June 22nd, 2004.

[56] http://www.surfsleutel.nl. SurfSleutel, June 22nd, 2004.

[57] http://www.vlerick.be/news/more/2002/children_internet.htm. Press release: "Children and the Internet" (2002). Vlerick Leuven Gent Management School. September 15th, 2004.

[58] http://www.hotmail.com. September 22nd, 2004.

Children and Usability

[59] Bilal, D. (2000), *Childrens use of Yahooligans! Web search engine: 1. Cognitive, physical and affective behaviors on fact-based search tasks*, Journal of the American Society for Information Science, 51(7), pp. 646665.

[60] Bilal, D. (2001), *Children's use of the Yahooligans! Web search engine: II. Cognitive and physical behaviors on research tasks*, Journal of the American Society for Information Science and Technology, 52(2), pp. 118-136.

[61] Bilal, D. (2002), *Children's use of the Yahooligans! Web search engine. III. Cognitive and physical behaviors on fully self-generated search tasks* Journal of the American Society for Information Science and Technology, 53(13), pp. 1170-1183.

[62] Bilal, D. (2002), *Perspectives on children's navigation of the World Wide Web: does the type of search task make a difference?*, Online Information Review, 26(2), pp. 108-117.

[63] Bilal, D. and Kirby, J. (2002), *Differences and similarities in information seeking: Children and adults as Web users*, Information Processing & Management, 38(5), pp. 649-670.

[64] Boren, T. and Ramey, J. (2000), *Thinking aloud: reconciling theory and practice*, IEEE transaction on professional communication, 43(3).

[65] Donker, A. and Markopoulos, P. (2001), *Assessing the effectiveness of usability evaluation methods for children*, In Avouris, N., and Fakotakis, N., (Eds.) *Advances in human computer interaction*, I, Typorama Publications, Greece, ISBN-960-7620-18-6, pp. 409-410.

[66] Druin, Allison and Solomon, Cynthia. (1996), *Designing Multimedia Environments for Children.*

[67] Druin, A., B. Bederson, A. Boltman, A. Miura, D. Knotts-Callaghan and M. Platt (1999), *Children as our technology design partners*, *The Design of Children's technology*, A. Druin. San Francisco, CA, Morgan Kaufmann, pp. 51 - 72.

[68] Druin, A., *The Role of Children in the Design of New Technology*, University of Maryland, http://www.cs.chalmers.se/idc/ituniv/student/2003/portfolior/it3daje/public_html/webbplats/Anvandartest/Artiklar/The Role of Children in the Design of New Technology.pdf, August 2004.

[69] Gilutz, S. & Nielsen, J. (2002), *Usability of Web-sites for Children: 70 Design Guidelines*, Fremont, CA, Nielsen Norman Group.

[70] Hanna, L., K. Risden and K. Alexander, J (1997), *Guidelines for usability testing with children*, Interactions, 1997(5), pp. 9-14.

[71] Jaeger, P., Bertot, J., McClure, C. (2004) *The effects of the Children's Internet Protection Act (CIPA) in public libraries and its implications for research: A statistical, policy, and legal analysis*, Journal of the American Society for Information Science and Technology, 55(13), pp 1131 - 1139.

[72] Jain, H., Vitharana, P., and Zahedi, F. (2003), *An assesment model for requirements identification in component-based software development*, 34(4), ABI/INFORM Global.

[73] Markopoulos, P., Bekker, M. (2002), *How to compare usability testing methods with children participants*, In Bekker, M., Markopoulos, P., and Kersten-Tsikalkina, M. (Eds.) Interaction Design and Children, Shaker Publisher, ISBN 90-423-0200-3, pp. 153-159.

[74] Large A., & Beheshti, J. (2000), *The Web as a classroom resource: Reactions from the users*, Journal of the American Society for Information Science, 51(12), pp. 10691080.

[75] Large, A., J. Beheshti, and C. Cole. (2002), *Information architecture for the Web: the IA Matrix approach to designing children's portals*, Journal of the American Society for Information Science and Technology, 53(10), pp. 831-838.

[76] Large, A., Beheshti, J., & Rahman, T. (2002), *Design criteria for childrens Web Portals: The users speak out*, Journal of the American Society for Information Science and Technology, 53(2), pp. 79-94.

[77] Large, A., Beheshti, J., Nesset, V., & Bowler, L. (2004), *Designing Web portals in intergenerational teams: Two prototype portals for elementary school students*, Journal of the American Society for Information Science and Technology, Early View.

[78] Read, Janet, *HCI with children for students*, ChicCisem.

[79] Read, J, *Designing stuff for Children!*

[80] Rockman. (2002), *Designing for Kids in the Digital Age*

[81] Solomon, P. (1993), *Children's information retrieval behavior: A case analysis of an OPAC*, Journal of the American Society for Information Science, 44(5), pp. 245-264.

[82] Vessey, I. and Conger, S.A., *Requirements specification: Learning object, process, and data methodologies*, Association for Computing Machinery. Communications of the ACM, May 1994, 37(5), ABI/INFORM Global

[83] http://www.webliminal.com/search/search-web04.html. Directories and Virtual Libraries, July 6th, 2004.

[84] http://yahooligans.yahoo.com/content/buzz/. Yahooligans! Buzz Index. September 28th, 2004.

[85] http://www.icra.org/faq/server/. ICRA, May 25th, 2004.

[86] http://www.cogsci.princeton.edu/∼wn/. WordNet, a lexical database for the english language, June 6th, 2004.

Search engines for kids

[87] http://www.explorian.com/ned/indexplo/frame.htm.
Indexplorian, May 25th, 2004.

[88] http://www.ajkids.com. Ask Jeeves For Kids, May 25th, 2004.

[89] http://www.yahooligans.com. Yahooligans, May 25th, 2004.

[90] http://www.ithaki.net/kids. Ithaki for Kids.

[91] Gudmundsen J., *Sites help kids learn how to use Internet for research.* Special to the Mercury News. Posted on Wed, Nov. 20th, 2002.

[92] http://www.noodletools.com. NoodleTools, August, 2004.

[93] http://www.a1b2c3.com/free/aaa_ads/goochi.htm.
Google SafeSearch, July 9th, 2004.

Cognitive Aspects Children:

[94] Bandura, A. (1986), *Social foundations of thought and action: A social cognitive theory*, Englewood Cliffs, NJ, Prentice Hall.

[95] Berland, Gretchen K. et al. (2001), *Health Information on the Internet - Accessibility, Quality, and Readability in English and Spanish*, http://jama.ama-assn.org/cgi/reprint/285/20/2612.pdf, September 2004.

[96] Carroll, J.B., Davies, P., and Richman, B. (1971), *The American heritage word frequency book*, Boston, Houghton Mifflin, ISBN 0395135702.

[97] Malone, T. W., & Lepper, M. R. (1987), *Making learning fun: A taxonomy of intrinsic motivation for learning*, In R. E. Snow and M. J. Farr (Eds.), *Aptitude, learning and instruction. Volume 3: Conative and affective process analysis*, Hillsdale, NJ, Lawrence Erlbaum.

[98] Martins, B., and Silva, M.J., *Spelling Correction for Search Engine Queries, University of Lisboa*, http://xldb.fc.ul.pt/data/Publications_attach/spellcheck.pdf, September 2004.

[99] McLaughlin, H. (1962), *SMOG grading a new readability formula*, Journal of Reading, 22, pp. 639-646.

[100] Wright BD, Strenner AJ (1998), *Readability and Reading Ability*, Presentation to Australian Council on Education Research (ACER).

[101] http://www.childdevelopmentinfo.com/development/
language_development.shtml

[102] http://www.childdevelopmentinfo.com/development/piaget.shtml

[103] http://www.lexile.com. The Lexile Framework.

[104] http://www.up.univ-mrs.fr/wpsycle/documentpdf/documentpiolat/Publications/. Sigverone, Barbier. May 12th, 2004.

Interface Design:

[105] Frakes, William and Baeza-Yates, Ricardo. (1992) *Information Retrieval Data Structures & Algorithms*, Prentice Hall PTR.

[106] Baeza-Yates, Ricardo and Ribeiro-Neto, Berthier (1999), *Modern Information Retrieval*, Essex, Addison Wesley Longman Limited.

[107] Inkpen, K. (2001), *Drag-and-Drop versus Point-and-Click Mouse Interaction Styles for Children*, ACM Transactions on Computer-Human Interaction, 8(1), pp. 1 - 33.

[108] Kleinfeldt, S. and Baphna J. (2000) *A Commercial Perspective on Hypertext Search Results.* http://www.avaquest.com/InfoDoors/.

[109] Nielsen, J. (1999), *Designing web usability*, Indianapolis, New Riders Publishing.

[110] Insights from Human Factors International (2003), *Key Research Findings Related to User-Centered Design (2002-2003)*, UI Design Update Newsletter December, 2003. http://www.humanfactors.com/downloads/dec03.asp

[111] Uden, L. and Dix, A. (2000), *Iconic Interfaces For Kids On The Internet*, http://www.hiraeth.com/alan/.

[112] Shneiderman, B. (1998), *Designing the User Interface. Reading*, MA, Addison Wesley Longman.

[113] Weeda, R. and Kuipers, S. (2003), *Meiden, Jongens en het Internet: informatie voorziening voor ouders van gehandicapte kinderen.*

Information Retrieval:

[114] Allan, J., Leouski, A. and Swan, R. (1997) *Interactive Cluster Visualization for Information Retrieval.* http://ciir.cs.umass.edu/pubfiles/ir-116.pdf.

[115] http://www.acm.org/sigchi/chi95/Electronic/documnts/doctoral/sgh_bdy.htm

[116] Crouch, C. and Yang, B. (1992) "Experiments in automatic statistical thesaurus construction, Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval, p.77-88, June 21-24, 1992, Copenhagen, Denmark

[117] Grootjen, F. and Weide, van der T. (2004), *Conceptual Query Expansion.*

[118] Grootjen, F. (2004), *Pragmatic Approach to the Conceptualisation of Language.*

[119] OCLC, *The DDC 22 Introduction*, http://www.oclc.org/dewey/, Aug 24th, 2004.

[120] http://www-lib.nearnorth.edu.on.ca/dewey/ddc.htm. Dewey Decimal Classification System (13th Abridged), September 23rd, 2004.

[121] Hunter, D. (2000), *Internet filter effectiveness (student paper panel): testing over and underinclusive blocking decisions of four popular filters*, Proceedings of the tenth conference on Computers, freedom and privacy: challenging the assumptions.

[122] Leouski, A. and Croft, W. () *An Evaluation of Techniques for Clustering Search Results.* http://ciir.cs.umass.edu/pubfiles/ir-76.pdf.

[123] Papadopoulos, N. and Plexousakis, D. (2002) "The Role of Semantic Relevance in Dynamic User Community Management and the Formulation of Recommendations". CAISE 2002, LNCS 2348, pp. 200-215. http://citeseer.ist.psu.edu/papadopoulos02role.html.

[124] Qiu, Y. and Frei, H.-P. (1994). "Improving the retrieval effectiveness by a similarity thesaurus". Technical Report 225, ETH Zurich, Department of Computer Science. http://citeseer.ist.psu.edu/qiu94improving.html.

[125] Sibun, P. and Spitz, A. (1994) *Language Determination: Natural Language Processing from Scanned Documents.*

[126] http://cis.pasco.k12.fl.us/contentareas/rdg_tips.html#middleschool. District School Board of Pasco County, "Reading Information and Tips for Pasco Parents". October 2nd, 2004.

[127] http://www.cs.umd.edu/hcil/kiddesign/searchkids.shtml. SearchKids Digital Library for Children, University of Maryland, September 2nd, 2004.

[128] White, R., Jose, J., and Ruthven, I. (2004) "An Implicit Feedback Approach for Interactive Information Retrieval". To appear in Information Processing and Management. http://www.dcs.gla.ac.uk/ whiter/white.pdf