Bachelor Thesis

A zoomable topic-based browsing approach to support search and sense-making in on-line book repositories

submitted for acquirement of the academic degree of a
Bachelor of Science (B.Sc.)

Presented By

Mario Schreiner

University of Konstanz

Faculty of Sciences
Department of Computer and Information Science
Group of Human-Computer-Interaction

Reviewers:

Herr Professor Dr. H. Reiterer
Herr Professor Dr. D. Keim

University of Konstanz
Department of Computer and Information Science
D–78457 Konstanz
Germany

August 23, 2013

Abstract

Nowadays, the internet, growing document spaces and digital libraries lead to an increased complexity of search tasks. Modern researchers can not hope to read every available bit of information anymore. Instead, knowledge seekers need digital help to distinguish relevant from irrelevant search results. Subject of this work is the design of a visual information seeking system that finds new ways to increase the efficiency of search workflows. The resulting system integrates query formulation, browsing and sense-making into a unified application context and combines a topic-based approach with advanced multi-touch interaction and a visual representation of results.

Einleitung

Heutzutage führen das Internet, wachsende Dokumentenräume und digitale Bibliotheken zu einer steigenden Komplexität von Suchen. Moderne Forscher können nichtmehr darauf hoffen, jede verfügbare Information auch lesen zu können. Stattdessen benötigen Sie digitale Hilfestellung um relevante von irrelevanten Suchergebnissen unterscheiden zu können. Gegenstand dieser Arbeit ist das Entwickeln eines *visual information seeking systems*, das neue Wege erschliesst die Effizienz von Suchabläufen zu steigern. Das daraus entstehende System integriert die Schritte der Suchanfrage, des Browsings und des Sense-makings in einen einzigen Anwendungskontext und kombiniert einen themenbasierten Ansatz mit fortschrittlicher Multitouchinteraktion und einer visuellen Repräsentation von Suchergebnissen.

CONTENTS

FIGURES

LISTINGS

CHAPTER 1

INTRODUCTION

Computer-aided searching has grown tremendously in importance and popularity over the last decades. Besides the ever-increasing speed of computers, the internet is one of the main reasons for that. With its nearly unlimited amount of information and its rapid distribution it found its way in the everyday life of most people, and also in the everyday work of information seekers and researchers. Today, the internet is an integral part of the life and work of those people and often used on a day-to-day basis.

While, in the early days of the internet the problem has been to connect people and allow them to share information with what limited bandwidth was available, today the information-related problems shifted. The amount of information became so huge that it is impossible for a group of individuals to consume every available bit of information, even on just a single topic. Ways for using computers to find information and also to assess the relevance of those information therefore continue to grow in importance. This thesis takes a look at computer-aided search and visual information seeking systems that allow people, and knowledge seekers in particular, to access and explore large amounts of information and create knowledge from them. It will explore how those systems evolved over the last two decades and if current systems are suited for the tasks and challenges faced by modern researches. Furthermore, our own concepts and solutions for the problems a researcher faces in his work will then be shown and we will introduce a prototype that was implemented as part of the *Blended Library* at the University of Konstanz. With this prototype, we approach the problem of searching through large amounts of information and propose solutions on how to solve them. This prototype will then be taken to the test in a comparative study and its concepts will be evaluated.

1.1 Searching in the Blended Library

At the Human-Computer-Interaction group at the University of Konstanz, the *Blended Library* (BL) project[30, 6] creates a modern library room that integrates ordinary, book-based library work with modern, computer-aided technologies and blends those two worlds. We will take a closer look at the concept of the *Blended Library* and our role in it in Chapter 4 on page 31.

At the beginning of this bachelor work, the digital help that the BL offers to knowledge workers was examined and the question was how those knowledge workers could be further aided in their work. A lot of interesting concepts and prototypes were already available, for example *TwisterSearch*[72], a collaborative search system (see Section 2.12 on page 16); the *Augmented Shelf*[42], a real-world bookshelf that is enhanced with digital information via augmented reality; or the *Blended Shelf*[41], a virtual bookshelf that represents digital media with attributes known from the real-world. Additionally, there were some systems currently in development, for example a prototype to support interactive reading[10]; the *Integrative Workplace*, a prototype for augmented documents that makes it possible to use digital technologies like searching and annotation on real-world documents[24, 23]; or *FacetSearch++*[7], an advanced visual query formulation system using tangible tokens (see Section 2.13 on page 17).

When exploring the possibilities of the BL and comparing the available projects to a knowledge workers workflow, a gap was noticed: Some systems regarded the work with individual documents and some systems regarded searching, with most of the search systems focusing on the step of query formulation. Support for browsing the search results, finding relevant results in large data spaces and making sense of documents to create knowledge was still limited. Of course, the ordinary library search of the University of Konstanz[56] and ordinary web search engines like Google[32] were available to workers, but the question came up if there couldn't be a better way to help users in finding and exploring relevant documents - a way specifically tailored towards the needs of modern knowledge seekers in a library.

After all, an integral part of library work is the time spent searching. This includes the search for books but also the search for documents and information on the internet, and of course putting the search results together to eventually create new knowledge. In this environment, with students and knowledge workers as the main audience, the searcher often has only a broad idea of what to look for. A student researching a topic doesn't necessarily look for "that one book" but is rather faced with the task of deciding which books to pick from a whole library of available material. This often makes searching by traditional means hard and a lot of time goes into reading materials that are discarded as irrelevant afterwards and then refining the search to get better results. In the bachelor project created prior to this thesis, a visual information seeking system was created that tries to address those issues with focus on the library context. To do that, current systems and available research on the topic of visual information seeking systems were evaluated and a number of important aspects and problems of such modern search systems were identified. Furthermore, the search process itself was examined: What is part of a search and which phases are problematic in current systems? With that knowledge, a visual information seeking system was implemented to address the issues and aid knowledge workers in the *Blended Library.* To do that, we propose a turn away from some of the concepts found in traditional search systems. For example, with large result sets, the need to replace the result visualisation of sequential, textual lists of documents came up. With those textual lists, the user can neither explore all the returned documents nor find the documents in the result set relevant to his research if the ranking algorithm does not put them at the top of the list. This results in a loss of information. After looking at existing research, we developed a topic-based approach as the solution. It gives the user a more abstracted view over large result sets and the user can then explore the data space and filter it implicitly by deciding to look only at the parts of the result set that are of interest. In the process, more and more detail is revealed until the user eventually explores individual documents. This will help the user in breaking down large result sets and finding relevant documents. We also implemented a visual representation of search results. Objects and actions are displayed visually on a two-dimensional information landscape, allowing the user to make use of spatial arrangement and zooming. Furthermore, besides improving individual steps of the search, we propose an integrated solution for searching. Knowledge workers often have to switch context during their work. Different digital and real-world systems for searching, note-taking, annotation and sense-making need to be managed by the user, and data needs to be constantly transferred between them, which consumes cognitive capacity. Using an integrated search system that uses a single interaction style and UI concept, and where all data is constantly available, would allow the user to focus on his task. Since covering all the steps of a search in a bachelor's project is out of scope, we focused on the parts we identified earlier as having limited support in the *Blended Library.* Our prototype will cover the browsing of large result sets and sense-making of found documents. We did design the system to work as seamlessly as possible with other BL systems, for example by using similar interaction. This reduces the cognitive effort in switching between systems in the BL. Furthermore, the systems could be combined in the future to form a single search system that aids the user in a completely integrated fashion.

In the end, our prototype will make it easier for a knowledge worker in the *Blended Library* to explore even very large data spaces, finding relevant documents, making sense of them and eventually being creative and create knowledge.

1.2 Outlook

In the following thesis we will take a look at different approaches to search by exploring the vast topic of visual information seeking systems in Chapter 2 on the next page. In that chapter we will also explore the theoretical concepts behind those systems and identify common concepts and problems that can help us in creating our own system. Then, we will look at the theoretical concepts of the search itself, the process and different phases of search and how they are implemented in modern search systems in Chapter 3 on page 20. We will also explore how today's library workers search, the concept of sense-making and why it is so important for the process of search. In Chapter 4 on page 31 we will then apply what we learned and propose a visual information seeking system as part of the *Blended Library.* We will propose an integrated system that helps knowledge workers in searching, browsing and making sense of the results. The system will be tailored towards library work and the search process conducted by library workers to aid them in their search for knowledge. That system will then be put to test in a real-world study in the library of the University of Konstanz in Chapter 5 on page 58. Finally we will conclude and take a look at possible further work in this area in Chapter 6 on page 64. The very deserved thanks to the people that made this thesis possible will be given in Chapter 7 on page 68.

CHAPTER 2

VISUAL INFORMATION SEEKING & SEARCH SYSTEMS

In this section we will take a look at past and existing visual information seeking systems (VIS systems / VISS) or other search systems. We will explore the theoretical concepts behind them and the way they approached the problem of visualising large amounts of information. We will also take a look at how we could apply what we learned here to our own approach of a visual information seeking system, which we will present in Chapter 4 on page 31.

Visual information seeking systems are an age-old topic in computer science. Even before the internet, but especially since then, the problem of a growing number of information and the problem of how to present that information, so the user can extract knowledge from it, is known. As Ahlberg and Shneiderman put it, "the trickle of what we call information has grown into a river, and some say a flood"[3]. Similar, there is almost unlimited research about the topic (ironically, VIS systems could help solve the problem of researching them). This section will present some of the most important or most influential to our work.

2.1 HomeFinder

Williamson and Shneiderman tried to approach the information flood problem back in 1992, when they introduced *dynamic queries* in their *HomeFinder* prototype[86]. They said that one of the major problems with database systems is the need to formulate a complex query (such as an SQL expression) in order to get a useful result. In order to solve this, they proposed direct manipulation interfaces. These interfaces offer a continuous representation of the search results together with visual interface elements for manipulating them. As we know today, their proposal of such interfaces succeeded: In modern searches, techniques like *query preview* and filters are common and a user rarely has to enter queries in SQL or similar low-level languages. Nonetheless, this problem is not not solved yet, as we will discuss in Section 3.3 on page 26, when we explore the implementation of the search process in modern search systems. Although the query languages became less complex and more tailored towards non-expert users, the user still needs to translate his information need into some kind of query - for example the correct keywords of a web search. This need to formulate a query can not be underestimated as a hurdle for the user to get to the desired results - reducing this hurdle by making it easier to formulate a query or by making it easier to refine the search results is highly desirable in a VIS system.

2.2 Pad

In 1993, Perlin and Fox introduced *Pad*[58]. It will be seen later that *Pad* introduces a lot of the concepts that were picked up and refined years later at the Human-Computer-Interaction (HCI) group of the University of Konstanz, and many of the concepts introduced with *Pad* are also a part of our prototype system. Pad tries to make use of the humans natural spatial way of thinking. *Pad* is "an infinite two dimensional information plane [...] Objects are organized geographically; every object occupies a well defined region on the Pad surface"[58, p. 1]. *Pad* additionally introduces the concept of portals - digital, movable magnifying glasses. A portal can enlarge items or render them differently, for example the items under a portal could be rendered with a specific visualisation. Furthermore, the ability to have multiple portals allows the user to view different parts of the landscape with different granularity - a concept later seen in a similar form as the "degree of interest" in the *HyperGrid*[36] or the *SuperTable*[43]. Portals will also be seen again in the ZOIL framework[25, 89]. *Pad* also introduces the concept of semantic zooming. Semantic zooming allows object to be rendered depending on their current magnification, or size. This means an object can render different content depending on how much space it occupies on the screen. For example, a document could show only its title when it is small. When it is enlarged, the document could show its abstract and when enlarged further, reveal its full text. Semantic zooming is one of the main concepts of the ZOIL framework and also of our prototype, which makes use of ZOIL. We will use semantic zooming to allow the user to get different views on the data, showing different dimensions, without having to leave the application. We will see more on that in Chapter 4 on page 31. With the concept of semantic zoom, *Pad* had a major influence on our system.

2.3 FilmFinder

Ahlberg and Shneiderman further studied VIS systems in 1994[2] and found out that there are "several user interface design principles that consistently lead to high levels of [user] satisfaction"[2]. The two main principles they found were direct manipulation and support for browsing. They also distinguished browsing from ordinary information retrieval: They said that browsing puts more emphasis on the ability to rapidly filter search results, progressively refine search parameters and continuously reformulate search goals. With those two principles they proposed a system called the *FilmFinder*[3, 2]. It supports *dynamic query filtering* to filter by title, actor, director, runtime or MPAA movie rating and by that supports the principle of rapidly adjusting query parameters. They combined those dynamic queries with a *starfield display.* A starfield display is a scatterplot that supports selection and zooming of plot points. This was combined with *tight coupling.* Tight coupling describes the logical coupling of interface components. For example, if the user filters the movies in *FilmFinder* by year, so that only movies before 1935 are displayed, only actors that have played in movies of that time period should be selectable. Including their proposed design principles seems desirable to us - exploring a data space by browsing and the ability to directly manipulate the data space, for example by selection and zooming, are still highly relevant and desirable attributes of a VIS system.

2.4 INSYDER and the SuperTable

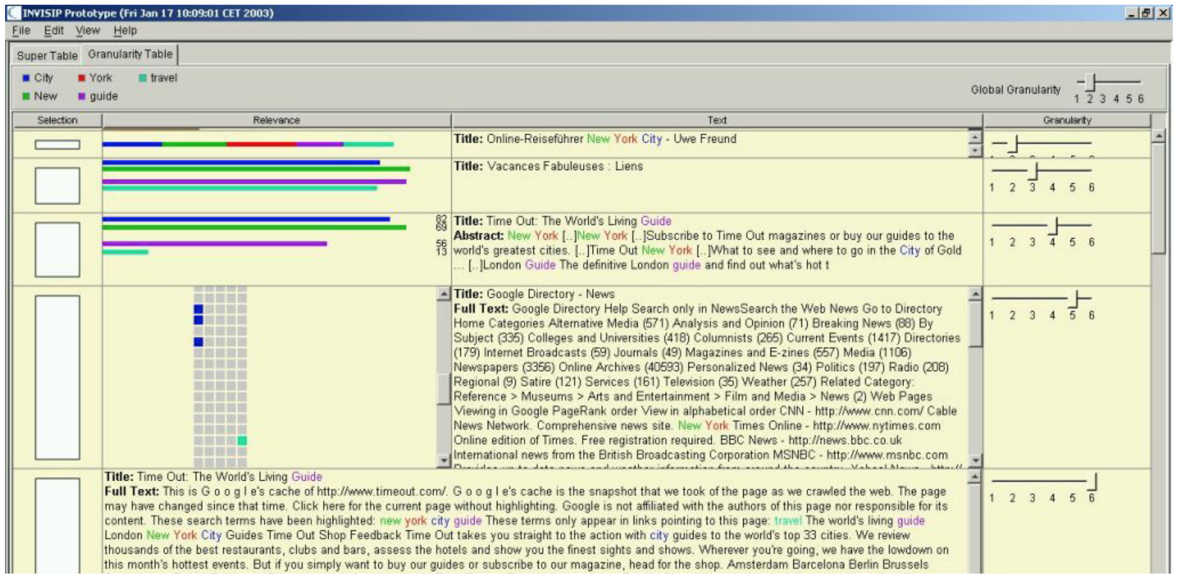
Looking at modern systems, it is interesting to see that the group of Human-Computer-Interaction at the University of Konstanz has a very long history of metadata browsers and visual information seeking systems, which strongly influenced our prototype.

Back in 2000, Mann and Reiterer presented INSYDER, an information assistant to retrieve business intelligence information from the web[68, 45, 67, 69]. They already saw some major problems with the information retrieval systems of that time: The question of how to express the information need, which we will take a look at in Section 3.3 on page 26, and the problem of how to best present the search results to the user, which we will examine in Section 3.4 on page 28, were just two of the major issues. INSYDER is an information assistant and Mann and Reiterer formed the term “sphere-of-interest (SOI)”[68, p. 2] to refer to different areas of interest of the user. While the SOI concept is an interesting one, it is not this part of INSYDER that is relevant to us - with the library context, the SOI of the users of our system is clearly defined. INSYDER is more than just the concept of SOIs, though: It proposes different agents that help the users in searching within the system. For example, it features an agent that helps the user in formulating the query. Different solutions are proposed to achieve this, for example the use of large textfields[78] or query expansion[79, 84]. INSYDER also features visualisation agents that help the user to analyse documents by visualising them in various ways. Mann and Reiterer state this is “a substantial assistant in [the] review of [the] results phase”[68, p. 3]. They used multiple visualisations in INSYDER, each delivering a different view on the result set, from a general overview of the results (for example via a scatter plot visualisation) down to individual documents (INSYDER offered the possibility to view document thumbnails). In our prototype, we took that idea and developed it further with the possibilities of modern technologies. Our system will offer different visualisations for different levels of detail by making use of semantic zooming, allowing the user to move quickly between different levels of detail without loosing context.

In fact, a redesign of INSYDER by Limbach et al. in 2002[43], supports this approach: They conducted a user study and found out that users often stuck to the simple table visualisation, simply because switching between different visualisations was confusing to them. The solution proposed by Limbach et. al was combining the different visualisations into one. They proposed a component called the *SuperTable*[43, p. 6] that combined all previous visualisations into one. The *SuperTable* inspired much of the semantic zooming capabilities offered in our system: Combining different levels of detail, with an easy to understand overview showing little detail at first, and multiple levels of more detailed information available for exploration via *details-on-demand*[74].

2.5 INVISIP and VisMeB

An EU-funded project called INVISIP[28] used the *SuperTable* approach to facilitate a metadata browser in a geo-related domain. Based on INVISIP, the *VisMeB* (Visual Metadata Browser)[44, 66, 40, 52] was presented in the same EU project. *VisMeB* works domain-independent and can be used regardless of if it is a web search, geo-related search, library search or something else. *VisMeB* uses a combination of *SuperTable* and Scatterplot as the main visualisation, which was already proposed in the redesigned INSYDER. The evaluation conducted shows again that the overview first, details-on-demand approach used in the SuperTable seems to work well for displaying result sets.



Title: Online-Reiseführer New York City - Uwe Freund

Title: Vacances Fabuleuses : Liens

Title: Time Out: The World's Living Guide
Abstract: New York [..]New York [..]Subscribe to Time Out magazines or buy our guides to the world's greatest cities. [..]Time Out New York [..]What to see and where to go in the City of Gold ... [..]London Guide The definitive London guide and find out what's hot t

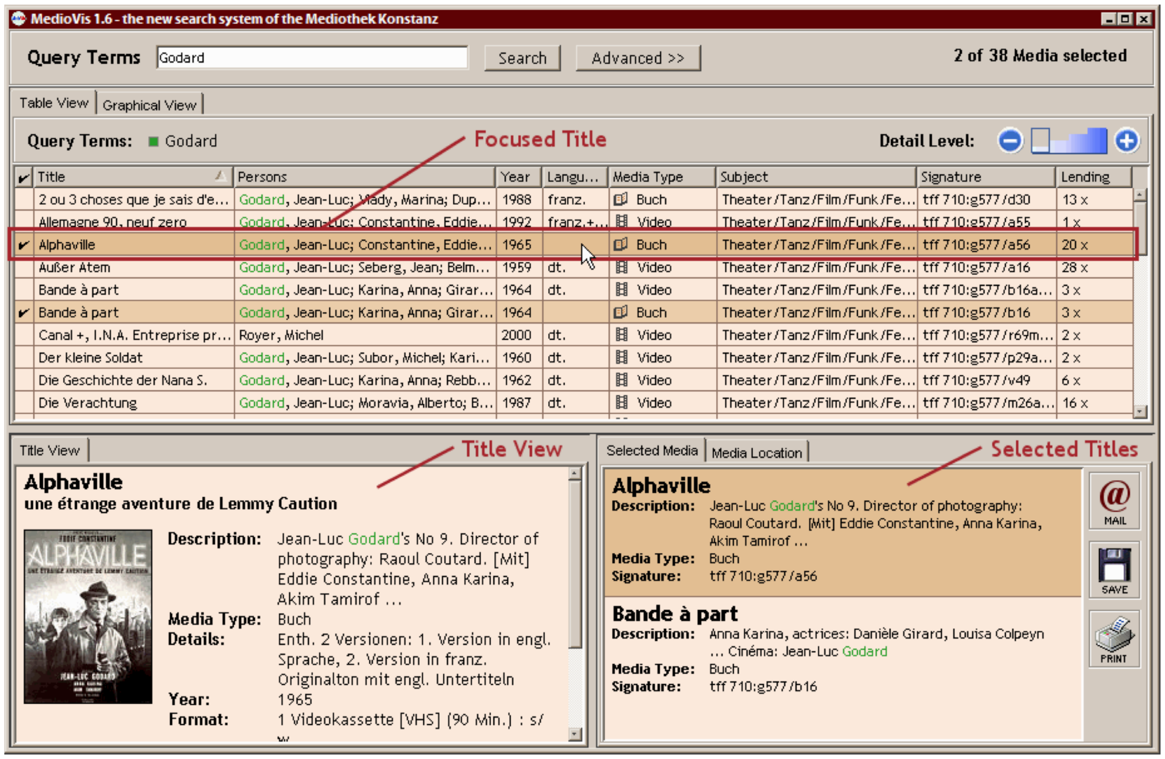
Title: Google Directory - News
Full Text: Google Directory Help Search only in NewsSearch the Web News Go to Directory Home Categories Alternative Media (571) Analysis and Opinion (71) Breaking News (88) By Subject (335) Colleges and Universities (418) Columnists (265) Current Events (1417) Directories (179) Internet Broadcasts (59) Journals (49) Magazines and E-zines (557) Media (1106) Newspapers (3356) Online Archives (40593) Personalized News (34) Politics (197) Radio (208) Regional (9) Satire (121) Services (161) Television (35) Weather (257) Related Category: Reference > Museums > Arts and Entertainment > Film and Media > News (2) Web Pages Viewing in Google PageRank order View in alphabetical order CNN - http://www.cnn.com/ Cable News Network. Comprehensive news site. New York Times Online - http://www.nytimes.com Online edition of Times. Free registration required. BBC News - http://news.bbc.co.uk International news from the British Broadcasting Corporation MSNBC - http://www.msnbc.com

Title: Time Out: The World's Living Guide
Full Text: This is G o o g l e's cache of http://www.timeout.com/. G o o g l e's cache is the snapshot that we took of the page as we crawled the web. The page may have changed since that time. Click here for the current page without highlighting. Google is not affiliated with the authors of this page nor responsible for its content. These search terms have been highlighted: new york city guide These terms only appear in links pointing to this page: travel The world's living guide London New York City Guides Time Out Shop Feedback Time Out takes you straight to the action with city guides to the world's top 33 cities. We review thousands of the best restaurants, clubs and bars, assess the hotels and show you the finest sights and shows. Wherever you're going, we have the lowdown on this month's hottest events. But if you simply want to buy our guides or subscribe to our magazine, head for the shop. Amsterdam Barcelona Berlin Brussels

Figure 1: The *SuperTable* in *VisMeB*. Each row has a different granularity, giving different information on each row. Source: [52]

2.6 MedioVis

Building on the evaluation of INSYDER, INVISIP and *VisMeB*, Grün et al. presented the user-centred library metadata browser *MedioVis* in 2005[27]. *MedioVis* and its successor, *MedioVis 2.0*[29], which will be discussed later in this chapter, both proved to be a big inspiration to our VIS system. *MedioVis* further emphasises the importance of details-on-demand and uses *multiple coordinates views*[53, 54] to parallelise the different stages of search (see Figure 2). They also point out the flexibility needed in keyword searches, without the user being forced to adhere to internal formatting structures of the database. They developed a single-input search that searches across multiple data to solve this. We will see in Section 4.4.1 on page 35 that we developed a very similar type of search for our system, which works in a cross-metadata single-input fashion. An thorough evaluation of *MedioVis* showed that both in regard to completion time of tasks and subjective user rating, *MedioVis* was clearly better than the media search system KOALA, which was the default local catalogue search for the University of Konstanz at that time.



✓	Title	Persons	Year	Langu...	Media Type	Subject	Signature	Lending
	2 ou 3 choses que je sais d'e...	Godard, Jean-Luc; Vlady, Marina; Dup...	1988	franz.	Buch	Theater/Tanz/Film/Funk/Fe...	tff 710:g577/d30	13 x
	Allemagne 90, neuf zero	Godard, Jean-Luc; Constantine, Eddie...	1992	franz.+...	Video	Theater/Tanz/Film/Funk/Fe...	tff 710:g577/a55	1 x
✓	Alphaville	Godard, Jean-Luc; Constantine, Eddie...	1965		Buch	Theater/Tanz/Film/Funk/Fe...	tff 710:g577/a56	20 x
	Außer Atem	Godard, Jean-Luc; Seberg, Jean; Belm...	1959	dt.	Video	Theater/Tanz/Film/Funk/Fe...	tff 710:g577/a16	28 x
	Bande à part	Godard, Jean-Luc; Karina, Anna; Girar...	1964	dt.	Video	Theater/Tanz/Film/Funk/Fe...	tff 710:g577/b16a...	3 x
✓	Bande à part	Godard, Jean-Luc; Karina, Anna; Girar...	1964		Buch	Theater/Tanz/Film/Funk/Fe...	tff 710:g577/b16	3 x
	Canal +, I.N.A. Entreprise pr...	Royer, Michel	2000	dt.	Video	Theater/Tanz/Film/Funk/Fe...	tff 710:g577/r69m...	2 x
	Der kleine Soldat	Godard, Jean-Luc; Subor, Michel; Kari...	1960	dt.	Video	Theater/Tanz/Film/Funk/Fe...	tff 710:g577/p29a...	2 x
	Die Geschichte der Nana S.	Godard, Jean-Luc; Karina, Anna; Rebb...	1962	dt.	Video	Theater/Tanz/Film/Funk/Fe...	tff 710:g577/v49	6 x
	Die Verachtung	Godard, Jean-Luc; Moravia, Alberto; B...	1987	dt.	Video	Theater/Tanz/Film/Funk/Fe...	tff 710:g577/m26a...	16 x

Alphaville
une étrange aventure de Lemmy Caution

Description: Jean-Luc Godard's No 9. Director of photography: Raoul Coutard. [Mit] Eddie Constantine, Anna Karina, Akim Tamirof ...
Media Type: Buch
Details: Enth. 2 Versionen: 1. Version in engl. Sprache, 2. Version in franz. Originalton mit engl. Untertiteln
Year: 1965
Format: 1 Videokassette [VHS] (90 Min.) : s/

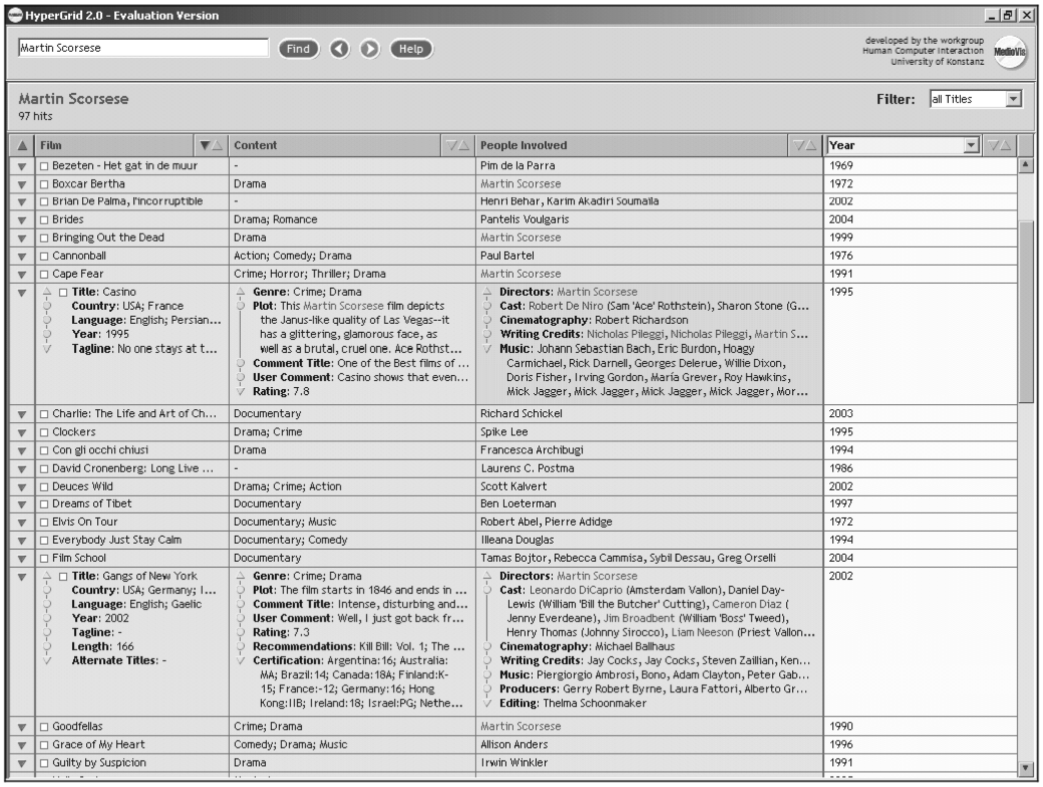
Alphaville
Description: Jean-Luc Godard's No 9. Director of photography: Raoul Coutard. [Mit] Eddie Constantine, Anna Karina, Akim Tamirof ...
Media Type: Buch
Signature: tff 710:g577/a56

Bande à part
Description: Anna Karina, actrices: Danièle Girard, Louisa Colpeyn ... Cinéma: Jean-Luc Godard
Media Type: Buch
Signature: tff 710:g577/b16

Figure 2: The main screen of *MedioVis*. It shows the concept of *multiple coordinated views* to show different views on the data space. In the center is the result list, in the bottom left is the individual title view and in the bottom right, selected titles are shown. Source: [27]

2.7 The ZoomWorld and HyperGrid

The plan to follow in the footsteps of the *SuperTable* and *MedioVis* was backed up in a paper by Reiterer et al. They built on their previous knowledge with metadata browser and VIS systems and explored the possibilities and traps of zoomable interfaces (ZUIs) in 2005[65]. They especially noted the book "The Humane Interface" by Jef Raskin[61] which presents the vision of the *ZoomWorld* where the user can navigate through an infinite two-dimensional landscape and content is placed on that landscape, regardless of the contents physical location (local or remote). The user navigates this landscape by panning and manages the granularity of details by zooming in or out. The zooming can go as far as looking at the actual object (for example, an Excel file) by zooming in far enough. Concepts like hyperlinks are completely gone in *ZoomWorld*. Raskin also says that such interfaces are superior to traditional, abstract interfaces, because they are closer to the characteristics of the human cognition. Reiterer et al. point out that, while Raskins vision of *ZoomWorld* is appealing, it does have its drawbacks: For example the turn away from anything standardised might lead to a break with conformance and expectations of users and because of that lead to low acceptance. Also, they point out that the user might be overwhelmed by the completely unguided, free navigation in the two-dimensional space. Furthermore, comparing contents that are far away in space might be difficult and not be achieved without further help. Reiterer et al. conclude that a combination of zoomable navigation and traditional interfaces might be a solution. As such a combination, they present the *HyperGrid*[36]. The *HyperGrid* combines traditional table and browser concepts with zoomable interfaces. It presents the user a familiar table structure but embeds the concept of zooming by giving the ability to semantically zoom into individual rows. The content of those rows changes depending on the zoom level (or "degree of interest (DOI)"), showing more information the deeper the user zooms in (see Figure 3). Additionally, the *HyperGrid* embeds multi-modality. Images and videos can be embedded directly into the table. Furthermore, to present external information, a browser window can be embedded directly into the table as well, making it possible to explore external information without the loss of context that a popup window would produce. This synergy between the concepts of *ZoomWorld* embedded in a traditional, familiar, tabular structure should lead to a higher conformity with expectations and therefore a higher acceptance by users.


HyperGrid 2.0 - Evaluation Version
Martin Scorsese
Find
Help
developed by the workgroup
Human Computer Interaction
University of Konstanz
MedioVis
Filter:
all Titles

Martin Scorsese
97 hits

Film	Content	People Involved	Year
Bezeten - Het gat in de muur	-	Pim de la Parra	1969
Boxcar Bertha	Drama	Martin Scorsese	1972
Brian De Palma, l'incorruptible	-	Henri Behar, Karim Akadiri Soumaila	2002
Brides	Drama; Romance	Pantelis Voulgaris	2004
Bringing Out the Dead	Drama	Martin Scorsese	1999
Cannonball	Action; Comedy; Drama	Paul Bartel	1976
Cape Fear	Crime; Horror; Thriller; Drama	Martin Scorsese	1991
Title: Casino **Country**: USA; France **Language**: English; Persian... **Year**: 1995 **Tagline**: No one stays at t...	**Genre**: Crime; Drama **Plot**: This Martin Scorsese film depicts the Janus-like quality of Las Vegas--it has a glittering, glamorous face, as well as a brutal, cruel one. Ace Rothst... **Comment Title**: One of the Best films of ... **User Comment**: Casino shows that even... **Rating**: 7.8	**Directors**: Martin Scorsese **Cast**: Robert De Niro (Sam 'Ace' Rothstein), Sharon Stone (G... **Cinematography**: Robert Richardson **Writing Credits**: Nicholas Pileggi, Nicholas Pileggi, Martin S... **Music**: Johann Sebastian Bach, Eric Burdon, Hoagy Carmichael, Rick Darnell, Georges Delerue, Willie Dixon, Doris Fisher, Irving Gordon, María Grever, Roy Hawkins, Mick Jagger, Mick Jagger, Mick Jagger, Mick Jagger, Mor...	1995
Charlie: The Life and Art of Ch...	Documentary	Richard Schickel	2003
Clockers	Drama; Crime	Spike Lee	1995
Con gli occhi chiusi	Drama	Francesca Archibugi	1994
David Cronenberg: Long Live ...	-	Laurens C. Postma	1986
Deuces Wild	Drama; Crime; Action	Scott Kalvert	2002
Dreams of Tibet	Documentary	Ben Loeterman	1997
Elvis On Tour	Documentary; Music	Robert Abel, Pierre Adidge	1972
Everybody Just Stay Calm	Documentary; Comedy	Illeana Douglas	1994
Film School	Documentary	Tamas Bojtor, Rebecca Cammisa, Sybil Dessau, Greg Orselli	2004
Title: Gangs of New York **Country**: USA; Germany; I... **Language**: English; Gaelic **Year**: 2002 **Tagline**: - **Length**: 166 **Alternate Titles**: -	**Genre**: Crime; Drama **Plot**: The film starts in 1846 and ends in ... **Comment Title**: Intense, disturbing and... **User Comment**: Well, I just got back fr... **Rating**: 7.3 **Recommendations**: Kill Bill: Vol. 1; The ... **Certification**: Argentina:16; Australia: MA; Brazil:14; Canada:18A; Finland:K-15; France:-12; Germany:16; Hong Kong:IIB; Ireland:18; Israel:PG; Nethe...	**Directors**: Martin Scorsese **Cast**: Leonardo DiCaprio (Amsterdam Vallon), Daniel Day-Lewis (William 'Bill the Butcher' Cutting), Cameron Diaz (Jenny Everdeane), Jim Broadbent (William 'Boss' Tweed), Henry Thomas (Johnny Sirocco), Liam Neeson (Priest Vallon... **Cinematography**: Michael Ballhaus **Writing Credits**: Jay Cocks, Jay Cocks, Steven Zaillian, Ken... **Music**: Piergiorgio Ambrosi, Bono, Adam Clayton, Peter Gab... **Producers**: Gerry Robert Byrne, Laura Fattori, Alberto Gr... **Editing**: Thelma Schoonmaker	2002
Goodfellas	Crime; Drama	Martin Scorsese	1990
Grace of My Heart	Comedy; Drama; Music	Allison Anders	1996
Guilty by Suspicion	Drama	Irwin Winkler	1991

Figure 3: The *Hypergrid*, where each row can have its own "degree of interest" that defines the information shown for that row. Source: [36]

2.8 The four design principles for VISS

In an evaluative paper, Gerken et al. summarise the last ten years of creating metadata browser and VIS systems at the HCI group at the University of Konstanz[25]. They identify four basic design principles that, according to them, should be adhered by a visual information seeking system. The four principles they propose are:

- **Support various ways to formulate an information need** – The requirement for a user to translate his information need into a search query which is understandable by the system is described as crucial but error-prone. By offering systems that allow a high flexibility in this step we ease that step for the user. We will take a closer look at the problem in Section 3.3 on page 26 and our solution to this in Section 4.4 on page 35.

- **Integrate analytical and browsing oriented ways of exploration** – In practice, users often switch between an analytical, goal-oriented search and discovery-driven browsing. VIS systems should support both of these ways of search to fully support the user. We will see that we offer support for both in our system in Section 4.6 on page 55.
- **Provide views to different dimensions of an information space** – This proved to be one of the major design principles of our system. We are adhering this design goal by providing an integrated way of getting an overview over the data space and then zooming in via semantic zoom until the user reaches individual documents. We will describe this in detail in Section 4.5 on page 39.
- **Make search a pleasurable experience** – One of the main goals of human-computer interaction is always to take the work out of tedious tasks and make them more enjoyable. With the use of modern technology, fluid interaction and alternative approaches we are aiming at this goal with our system - making the work with a system fun can never be underestimated, but is also one of the hardest goals to achieve in our opinion, especially in a prototypical system. This goal will be evaluated in a user study in Chapter 5 on page 58.

These design principles proved to be very important to our system. In particular, providing views to different dimensions and making search a pleasurable experience were major goals in the development of our system.

2.9 The ZOIL paradigm and the ZOIL framework

The concepts introduced so far were developed further to the point where the "novel user interface paradigm ZOIL (Zoomable Object-Oriented Information Landscape)"[89, p. 1] was proposed by Jetter et al. in 2008[36, 89, 38, 90, 39]. The ZOIL concept aims at "unifying all types of local and remote information items [...] in a single visual workspace as a replacement of today's desktop metaphor"[38, p. 1] and is "an application- and platform-independent UI concept"[38, p. 2]. ZOIL proposes an infinite two-dimensional space and all objects are part of that space, allowing to utilise the human abilities to recognise, remember and interpret spatial relationships. The ZOIL paradigm is based on five core principles:

- **Object-Oriented User Interface**: Regardless of their origin and form, information (for example from the web, file system, a database, ...) is packed into objects and the user can interact with those objects. The objects have the ability to derived from each other, so a common user interaction can be applied to all objects regardless of their type. If needed, specific interactions or views can be created for individual objects or groups of objects.
- **Semantic Zooming**: We already saw semantic zooming in *Pad* and a derived form of it in Raskin's *ZoomWorld*. In ZOIL, all objects are represented visually. Unlike in modern desktop systems, objects are placed on an information landscape which is not limited by screen size but is an infinite canvas. The user can move through the landscape by panning and zooming. When zooming, ZOIL uses semantic zoom: Objects can offer different content and functionality based on their current size on the screen. Jetter et al. say that "An important benefit of using semantic zooming to navigate the information space is the natural "feel" and the "intuitiveness" of zooming because of its similarity to the principles of the physical world"[38, p. 4].

- **Nested Information Visualisation**: Because information landscapes can grow extremely large quickly, a way of structuring them is needed. On a local hard drive, concepts like folders or tags are used for this. In ZOIL, portals can be created in the information landscape. For each portal a size, location, objects and visualisation type can be defined. For example, all images could be displayed as zoomable thumbnails on a map.
- **The Information Space as Information Landscape**: The information landscape of ZOIL tries to unify items of all kind on a single visual information workspace. Regardless of location (local or remote), type (image, speadsheet, web page, text message, real-world book, ...) or other factors, the information landscape is a space for all the users information. Portals and direct manipulation allow the user to manage this space and customize it. For example, often-used objects or portals can be kept at a prominent place for quick access.
- **Nomadic Cross-plattform User Interfaces**: A ZOIL landscape naturally scales to different display sizes. This also means that a consistent interaction can be used across devices when ZOIL is used. In the vision of ZOIL, it would be used across many devices and access a single information landscape, so a user has access to all his information from everywhere.

The vision of the ZOIL paradigm was manifested in the ZOIL framework[89], a C#/WPF based framework that incorporates the ideas and concepts of the ZOIL paradigm. The ZOIL framework also incorporates a large number of other important concepts of VIS systems, for example direct manipulation[86], details-on-demand[74], the four design principles for VIS systems by Gerken et al.[25] and a lot of the concepts introduced by *Pad* and the *ZoomWorld*. The ZOIL framework allows for the development of C#/WPF based applications that feature an infinite information landscape like the one proposed in the ZOIL paradigm. The landscape can be panned by a single-finger pan and zoomed by two-finger pinching. Objects can be placed on the landscape and have a size, position and are visually represented in a custom-defined way. Objects can be moved, resized and rotated freely on the landscape. Additionally, objects can have semantic zoom abilities - when the landscape is zoomed or the object is resized, its size on the screen increases or decreases. The object can render its content completely differently depending on its size on the screen. This also means the object can offer different functionality for different zoom levels - for example by displaying an advanced UI when zoomed into. Additionally, ZOIL allows for the creation of portals as they are proposed in the ZOIL paradigm. Because of the broad scope of the ZOIL framework, listing all the features and concepts would be out of scope of this thesis. Instead, we will now give some examples of research systems that have been implemented with the ZOIL framework to give an impression of the possibilities of the framework. It should be noted that this is by no means a full list of ZOIL based systems, only the ones most interesting to our research.

2.10 Media Seminar Room

The first system that made use of the ZOIL framework was the *Media Seminar Room*[8] by Hans-Christian Jetter with support by Mathias Heilig and a student team (Michael Zöllner, Mischa Demarmels, Stephan Huber, Oliver Runge, Benjamin Frantzen, Sebastian Rexhausen). The *Media Seminar Room* lets multiple students of a film seminar collaborate by accessing movie information and search, discuss and annotate those information with the goal of creating ideas and hypothesis. The room consists of two back-projected display cubes for pen input and a Microsoft Surface[48] for touch input and usage of tokens. A single ZOIL-based information landscape is provided on all devices simultaneously but each device can view a different part of the landscape. The landscapes displays all movies for the seminar and the landscape can be panned and zoomed into, revealing additional movie information via semantic zooming. At some point, the video stream of the movie is revealed.

The touch device is used to collaboratively alter the landscape, removing, rearranging, resizing and rotating movies. Changes made to the landscape on one device are immediately reflected on the other two devices. Additionally, *tangible lenses* can be placed on the tabletop. These are rectangular, transparent, real-world tokens. The part of the landscape below such a lens is tracked and sent to another display and visualised there in higher resolution and more detail. The lens can be moved to move through the data space on the additional display. This gives the illusion of moving a physical camera. Additionally, a physical *visualisation lens* can be put on the table. The content inside that lens will be visualised on the tabletop display but with an alternative visualisation, for example a scatterplot (see **??**). On the cubes, a part of the landscape can be selected with the pen by drawing around objects and the selection is then visualised. Additionally, users can use the pens to write on small paper notes made out of Anoto paper[1]. Those paper notes can then be put on the device and are identified and digitalised, creating a digital representation of the note in the ZOIL landscape. The *Media Seminar Room* demonstrates the abilities of the ZOIL landscape, tokens and real-world lenses and multi-device usage.

2.11 MedioVis 2.0

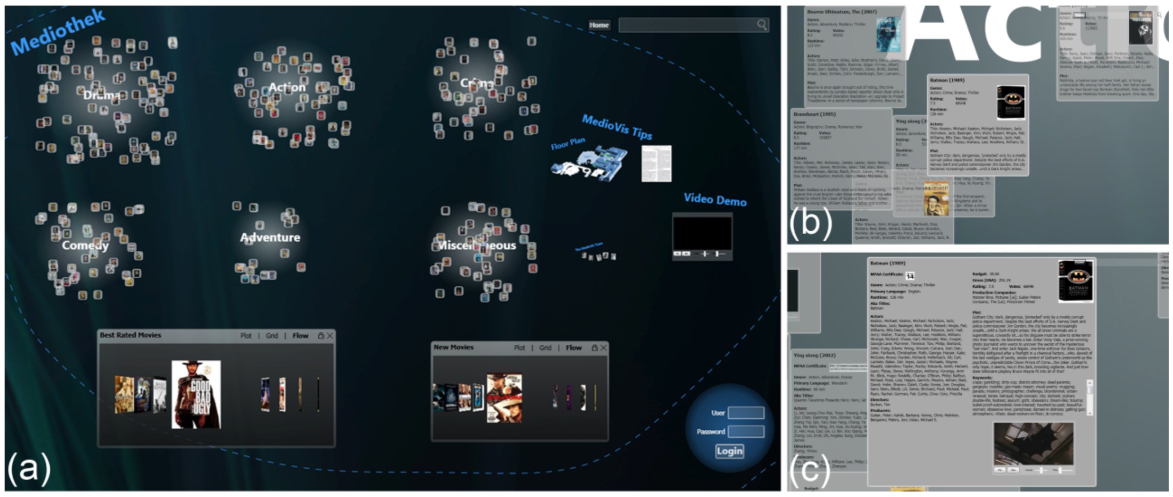


Figure 4: The main information landscape of MedioVis 2.0 (a) and two semantic zoom levels, each displaying a different granularity of movie information (b & c). Source: [64]

MedioVis 2.0[64, 29] was introduced in 2009 and built on the acquired knowledge from *MedioVis*, the ZOIL framework and the evaluations by Reiterer et al. and Gerken et al. While the first *MedioVis* served a very specific purpose, *MedioVis 2.0* was introduced as a *Knowledge Media Workbench* that supports the entire workflow of a knowledge worker. It put an emphasis on the visual representation of data and activities. Likewise, the concept of semantic zoom and with it the four design principles for VIS systems by Gerken et al. and the now often-mentioned concept of details-on-demand are implemented in the core of the system. Additionally, *MedioVis 2.0* puts emphasis on user experience and joy of use to achieve satisfaction with the user. Objects are placed on a ZOIL-based information landscape and visually represented. The application can basically support every kind of object, the prototype is connected to a movie database, showing the posters of movies and spatially clustering the movies by genre. The user then has the possibility to either filter by keyword search or go exploring on his own by panning and zooming. When the user zooms in further, the movies transform into more detailed descriptions, allowing the user to see metadata like title, plot, director and more. When zooming in even further, a trailer can be watched from right inside the application. This shows the concept of semantic zooming, multi-modality and staying in a single context, all properties we desire in our system. Additionally, *MedioVis 2.0* allows to use the concept of portals to get alternative views on the data space. The user can select a region of the space with a lasso gesture and a portal for the contents of that region will be created. The user can select a visualisation to display the contents of that region, for example a cover-flow visualisation. *MedioVis 2.0* has more features, but the ones that inspired the creation of our system most have been illustrated. Especially the concept of an integrated workspace, the focus on a playful and joyful use and the integration of semantic zooming were inspiring to us.

2.12 TwisterSearch

TwisterSearch[72, 73] supports a collaborative search workflow according to Evans and Chi's Canonical Model of Social Search[20, 21] that we will explore ourselves as an important search model in Section 3.1.3 on page 22. *TwisterSearch* allows users to gather search requirements by writing terms on Anoto paper[1] and putting them on a tabletop device. The terms are digitalised and can then be grouped into semantic clusters. Via tokens, users can assign clusters to themselves or each other. If a user "owns" a cluster, that user is responsible for researching that topic. The keywords of that cluster are sent to the user's personal device and the user can conduct a private search for those keywords. Found information can be shared by sending them to the shared tabletop device. Additionally, metadata like trails to the result and more are saved. The results can then be collaboratively discussed and new searches can be conducted if necessary.

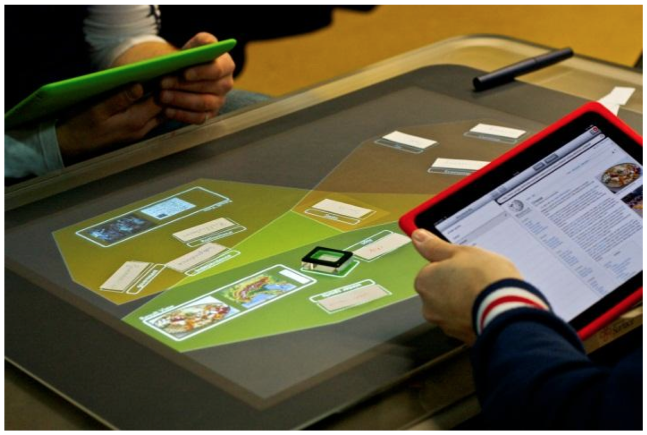


Figure 5: *TwisterSearch* during usage, showing the personal tablet of a user and the shared tabletop. A tablet-shaped token on the tabletop indicates that the cluster it is placed on is owned by the tablet of the same colour. Source: [73]

2.13 FacetStreams and FacetSearch++

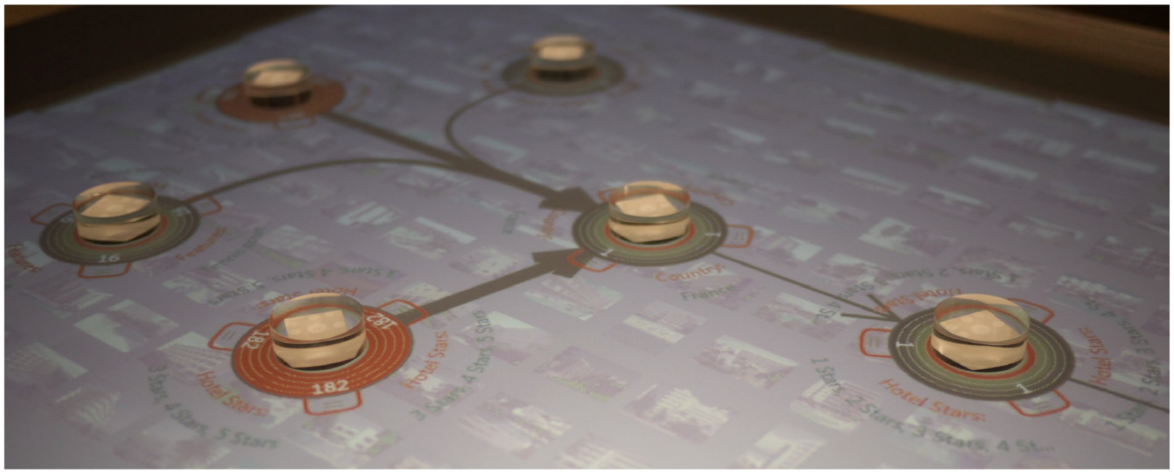


Figure 6: *FacetStreams* with multiple tokens. Directed streams connect the tokens. When a stream goes through a token, it is filtered by the tokens attached metadata and value range. Source: [37]

FacetStreams[37] allows for collaborative searching of databases. In the prototype, a hotel database is used. The application displays the hotels in a grid structure and allows to zoom and pan that grid to get information about hotels. Additionally, disc-shaped tangible tokens can be put on the tabletop allowing to select a metadata (*facet*) and value or value range. Tokens can then be linked to form a directed graph, or *stream*. Streams are a visual representation of a boolean search. When the stream passes a node it is filtered by the selected value range of the selected metadata of that token. Branching and merging streams allows for creating complex boolean queries implicitly, without a deeper understanding of boolean logic. This concept allows even novice users to build complex queries and make use of advanced motor skills by using tangible tokens. Also, the visual representation and easy modifiability of the streams encourage users to rapidly and often change their query.

Two user studies evaluating the concept of *FacetStreams* confirm that users perform equally good with the novel interface of *FacetStreams* as with an ordinary web interface. Additionally, the second study shows that non-expert users can build very complex boolean queries, even without excessive training phases.

FacetSearch++[7] is currently being developed as the successor to *FacetStreams*, building on the same concepts and interaction techniques. We will see later that *FacetSearch++* became one of our supported methods of query formulation and is therefore an integral part of this project.

2.14 Summary

We saw that there is a lot of research in the area of visual information seeking and search systems. It has to be stressed again that this was just a small excerpt of the available research, the part that was important or inspiring our prototype system. In conclusion, there is a multitude of available systems but most of them share certain key concepts that stand out and that we took as the most important for the creation of our visual information seeking system. We now know the importance of the ability of users to express their information need - we will go into depth on this in Section 3.3 on page 26.

We saw the *HomeFinder* that supported *dynamic queries*, *query expansion* and *filtering* as example methods to support this step.

We also learned the importance of *direct manipulation*, good support for *browsing* and *tight UI coupling* with the example of the *FilmFinder*.

We talked about *Pad*, a very early vision of an object-oriented way of expressing information on a two-dimensional landscape. Pad also introduced the key concept of *semantic zooming.* A lot of the concepts introduced by *Pad* were later picked up by other systems in a refined and modernised way and are still found in our prototype system.

We also looked at the long history of metadata browser and VIS systems at the HCI group at the University of Konstanz. Beginning almost fifteen years ago, different systems have been developed, refined, evaluated and, based on the gained knowledge, new systems have been developed, refined, evaluated, ... We learned about INSYDER and that even back then, the problems of expressing the information need and displaying the search results in a useful way, that goes beyond mere textual lists, were known. INSYDER also proposed some solutions to these problems - one key concept that is still highly relevant in modern systems being the integrated support of multiple dimensions of the data space. While INSYDER first tried to solve this by offering multiple visualisations and then was redesigned to incorporate the *SuperTable*, this concept was further refined over the years to the point were ZOIL approached this via the semantic zoom in modern years. An evaluation of INSYDER supports the importance of this concept. The *SuperTable* includes another key concept - the concept of *details-on-demand* first introduced by Shneiderman in 1996 with his mantra "overview first, zoom and filter, then details-on-demand"[74].

We then learned about the EU-funded INVISIP that used the *SuperTable* to bring the metadata browser to a geo-related domain and about *VisMeB*, an offspring of INVISIP. While INVISIP was domain-bound, *VisMeB* worked domain-independent and both strengthened the importance of the details-on-demand approach even further.

Then we learned about *MedioVis*, which parallelised the different stages of search with *multiple coordinated views.* In *MedioVis*, we also learned the importance of a *flexible keyword search.* It solved this by introducing a single-input search.

Reiterer et al. evaluated the past systems, especially in regard to the advantages and pitfalls of zoomable user interfaces (ZUIs). We learned about Jef Raskin's vision of the *ZoomWorld*, a two-dimensional, spatial interface where objects are represented visually and panning and zooming is used to move through the data space. Reiterer et al. also evaluated the dangers of this concept and introduced *HyperGrid* as a compromise, which tried to bring together the best of both worlds[70] by integrating the concepts of "degrees of interest (DOIs)", semantic zoom and multi-modality into the familiar environment of a table. The *HyperGrid* was supposed to increase user acceptance and expectation conformity while still offering much of the advantages of the introduced concepts.

Another evaluation was done by Gerken et al. and they introduced four major design principles for visual information seeking systems: Good support for expressing the information need, integrating analytical and browsing exploration, providing different views on the information space and making the usage of a system a pleasurable experience. We took those principles as goals for the system we want to create.

A manifestation of those principles was the ZOIL paradigm, a UI concept that tries to unify all different types of objects on a two-dimensional landscape. Objects are represented visually and the user can move on the landscape by panning and zooming. Semantic zoom offers the ability to show different dimensions of individual objects, eliminating the need to change contexts during the exploration of a data space. Additionally, the use of a two-dimensional landscape makes use of the very good human spatial perception abilities. We then saw different systems that made use of the ZOIL framework, a manifestation of the ZOIL paradigm in C#/WPF with multi-touch support. First was the *Media Seminar Room*, a room that allowed for the exploration of movies with a multitude of means, including semantic zoom, a shared landscape visible on multiple devices, tangible lenses, visualisation lenses, pen support and more. Then came *MedioVis 2.0*, a Knowledge Media Workbench that supported the entire workflow of a knowledge worker. It put emphasis on details-on-demand, semantic zoom, an integrated solution of search systems and joy in usage. We then looked at *TwisterSearch*, a collaborative search systems that allowed users to brainstorm ideas and then assign tasks to individuals. The tasks could be solved on personal devices, whereas the results could then again be displayed on a shared device and discussed. Lastly we looked at *FacetStreams* and *FacetSearch++*, which allow for the visual creation of complex boolean queries without a deeper understanding of boolean logic by using the concept of streams. An evaluation proved the concepts and we will see later that we implemented support for *FacetSearch++* in our prototype as an advanced mean of query formulation.

Furthermore, we conclude that using the ZOIL framework ourselves might help us in achieving our goals. As we have shown, it already provides a solid framework for a lot of the concepts we discovered to be important and it will additionally help us to integrate our system with other systems in the Blended Library. We will talk in detail about why we have chosen to use the ZOIL framework in Section 4.3.2 on page 34.

CHAPTER 3

SEARCH

After looking at the multitude of VIS and search systems available, this chapter will deal with the process of search itself. In general, it can be said that, when talking about search, one talks about a very broad field: There are many different approaches to computer-aided search, many different people using search engines for many different reasons and a lot of different requirements to a search system. Supporting people to find the population of a city on the internet can be vastly different from helping someone to find information about a movie in an online database or helping people researching a topic in their local library.

In the following, we will first take a look at models that help us grasp the broad process of search and determine how people search. We will then look at how computer-aided search evolved over the years and explore the different phases of a search in detail. We will discover how those phases are implemented in modern search systems and where there might be room for improvement. Finally, we will conclude what we learned from this and how it might influence our own visual information seeking system.

3.1 Theoretical models of search

3.1.1 Shneiderman's Framework for Mega-Creativity

In 1998 Shneiderman proposed the so-called genexes - generators of excellence[75, 76]. Shneiderman introduced genexes as families of tools to support creativity and innovation in a certain domain. With genexes, he wanted to help people in being creative and achieve something extraordinary with the help of digital tools:

> In looking to the future, we might again transform society by building genexes – generators of excellence – to support creative exploration of ideas. Thesauri are to words, as genexes are to ideas. [75, p. 1]

Shneiderman also puts emphasis on the good design and integrated fashion such tools should have in order to support these tasks. The concept of genexes is an important model to us, as the system we want to build can be considered a genex - an integrated collection of tools for library workers to find information and by that, hopefully, create knowledge and innovation. The concept of genexes is built on four foundational believes which Shneiderman lists. From those four believes, Shneiderman proposes a four-phase model of creativity[75]. Each believe is tied to one phase of the model and Shneiderman also lists example genexes that support each phase:

- **New knowledge is built on previous knowledge**: This is mapped to the "Collect"-phase where information is collected from existing knowledge. Genex tools that support this can be digital libraries, search services, dynamic queries, information visualisation or multimedia searches.
- **Powerful tools can support creativity**: This is mapped to the "Create"-phase where innovations are created using advanced tools. Example genex tools for this phase are user interface builders, simulations, models, macros, art, ...
- **Refinement is a social process**: This is mapped to the "Consult"-phase where the searcher consults with peers or mentors. Example tool are newsgroups, presentations, groupware or similar tools.
- **Creative work is not complete until it is disseminated**: This is mapped to the "Disseminate"-phase where results are spread widely. Example tools to support this phase are email, electronic publications, E-communities and more.

Shneiderman says that building systems that respect these four phases can help users in creating something really novel, innovative and help them in their creative work to exceed expectations. When building a visual information seeking system that is tailored towards knowledge workers it therefore seems desirable to incorporate Shneiderman's model. In the scope of this bachelor project and thesis, we focused on the phases "Collect" and "Create". We left room to include support for consultation, collaboration and sharing.

Shneiderman did publish a revised version of his model. He revises the previous four phases "Collect, Create, Consult, Disseminate" to the following *Framework for Mega-Creativity*[76, 77], or "four-phase genex framework"[76, p. 7]:

- **Collect**: learn from previous works stored in libraries, the Web etc.
- **Relate**: consult with peers and mentors in early, middle and late stages.
- **Create**: explore, compose, and evaluate possible solutions
- **Donate**: disseminate the results and contribute to the libraries

Transferring what we said about the original model to the *Framework for Mega-Creativity*, our system will still focus on supporting the "Collect" and "Create" phases and leave the "Relate" and "Donate" phases to future work. An important fact Shneiderman mentions with his framework is that it is not to be taken linearly but is an iterative process. This means that phases might occur in different order or earlier phases might be revisited one or multiple times during the search process.

3.1.2 Shneiderman's four-phase framework for textual search

Shneiderman also proposed another model, the four-phase framework for textual search[78]. Shneiderman talks about the problem of interfaces for textual searches. He says those interfaces are often confusing and hide key features, which leads to zero-hit or too-many-hit results, confusing the user. Another source of confusion can be inconsistencies between the interfaces of different searches. To satisfy users, Shneiderman then proposes his framework to make it easier for UI designers to create good and consistent search interfaces. The four phases proposed by Shneiderman are:

- **Formulation**: This phase contains a lot of user decisions like where to search, which fields to search, what to search for (the query) and what variants of a text to accept. Shneiderman describes this as the most complex phase of the process. Decisions like if the user wants to use boolean operators are also made in this phase and different systems treat these differently. The way a system treats different kinds of queries can be crucial: For example, a lot of systems look like they accept natural language, but queries like "bees and not honey" are actually searching for "bees honey", because the system removes stop words. The way a system treats queries, if it accepts only unstructured text, boolean queries or other kinds of queries is an important design decision.
- **Action**: The user starts the search either explicitly or implicitly. For example, users might click on a search button (explicit action) or the usage of dynamic queries might show the search results while the user is typing (implicit action). Shneiderman seems to prefer the implicit approach, but also notes that this might not always be possible due to available bandwidth, database speed, processing power or other factors.
- **Review of results**: Shneiderman talks about the problem of finding relevant results in a result list of even only a few hundred results, as they often occur in modern systems. He proposes alternatives to traditional, textual list results. He says a lot more can be done with modern visualisations to aid the user in finding relevant results, especially by visually representing results. He also says that the system can provide helpful messages where necessary that show the user why and how a certain result was found.
- **Refinement**: The system should support query refinement by supporting successive queries and other helpful feedback. A history of searches might be helpful as well. Shneiderman also proposes the ability to save and share searches.

We take these phases as the first search-tailored framework we discovered. The four phases described by Shneiderman will be backed up in the following sections. While other authors might differ in details on how search works, the general framework of Shneiderman - "Formulation, Action, Review of results, Refinement" - is largely shared by other researchers. The advices Shneiderman gives in his framework, especially in regard to good search UIs, will certainly be helpful in creating our visual information seeking system.

3.1.3 Evans and Chi's Canonical Model of Social Search

Evans and Chi defined their Canonical Model of Social Search[20, 21], which shaped our understanding of how search works and how we can support the different phases of the search process. While it goes into more detail than Shneiderman's four-phase framework and focuses on the social aspect of search, in general both have the same view on search. Because of the importance of this model for our understanding of search, we will now go into more detail on the different phases of the model. In general, Evans and Chi say that search is not only the actual search, but begins before the user even approaches a search system and ends long after the user finished the actual act of searching. They therefore split their model into three general phases, before search, during search and after search.

Before Search

Before the actual act of search, Evans and Chi describe different actions the user can or must take in order to begin searching. Part of this is the *context framing* where the user defines his information need - what it is he wants to know and how he will approach the search. The user might also undertake the *requirement refinement* phase, where the information need is further refined, for example by exchanging with others (in the case of a student, this exchange might for example be with his tutor).

During Search

This phase describes the actual act of search and is therefore the largest phase in Evans and Chi's model. They describe this phase as "traditional information seeking and foraging activities"[20, p. 2] and three types of searches can occur here: *Transactional search*, *navigational search* or *informational search*:

- **Transactional search**: This defines a search where the user wants to extract information from a well-known location. For example, getting driving directions through a web navigational service or reading a certain paragraph in a well-known article.
- **Navigational search**: In the process of the navigational search the user navigates through often familiar, well-known locations until the desired content is found. When the content is found it is often instantly recognised as such and the search stops. Examples for a navigational search would be browsing a database for a specific entry or looking at bookshelves in the library for a specific book.
- **Informational search**: This is the most interesting search to us. Since we built our system in the context of the Blended Library and specifically for use by knowledge workers, we know that this is the kind of search that our users will most likely conduct. An important difference between the other kinds of search and the informational search is that the informational search is open-ended. The goal of this search is to gain "knowledge" about a certain topic. Even after knowledge was created, there might still be more - it is unknown to the searcher if all vital information were collected. For example, if a student researches a topic on the internet and finds an interesting webpage, the student can write down the information found there. After that, the student is free to stop or to continue his search and look for more information - both are vital options. When the user conducts an informational search, documents are collected and skimmed to asses their relevance for the task. If they are rated relevant they will be marked as interesting. This marking can, for example, be bookmarking in a web browser, printing, copying the file to another folder etc. In the end, the user ends up with a number of documents, the so-called "evidence files"[20, p. 2]. With those evidence files, the user now enters the sense-making phase, where documents are read, evaluated, uninteresting documents are discarded and interesting documents are annotated and managed. The managing can for example consist of building piles according to the topic of the documents, drawing a mind-map of their contents, creating cue cards etc. The goal of the sense-making phase is to encode the documents in a way that makes them easier to understand and, before everything else, makes it possible to extract new knowledge out of the documents. While Evans and Chi only dedicate a small amount of their attention to sense-making, we will take a closer look at what sense-making is and how we want to support it in Section 3.5 on page 29 and at our actual support for sense-making in Section 4.6 on page 55.

After Search

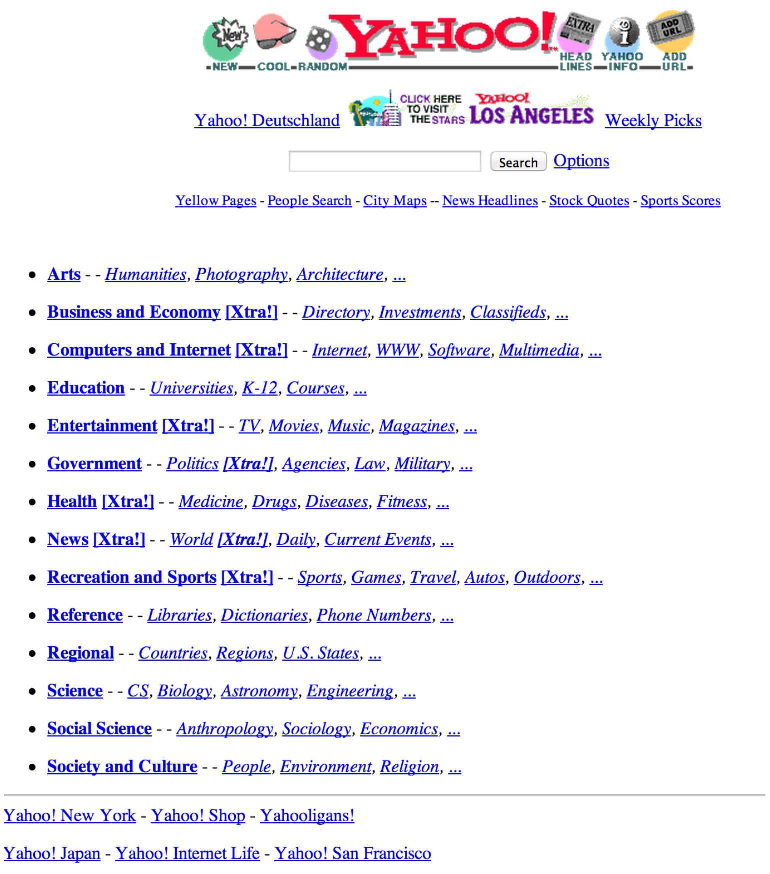
Evans and Chi emphasise that search does not stop after the "During Search"-phase. After the search, most of the time a *search product* is created from the knowledge acquired during the search. This product can, for example, be a presentation, an essay, a thesis, a paper or instructions on how to proceed with work and more. Additionally, that search product might be shared and distributed with others. This can consist of sending or showing it to others or make it available for a broader public (for example by creating a blog post). Evans and Chi point out that one can also distribute it to oneself, for example by simply saving it for future reads.

Lastly, materials and documents gathered during the search might be organised by the user for future reference. This can involve bookmarking, printing or writing a summary. While this is similar to the sense-making phase, it differs vastly in the goal and time it occurs. Sense-making occurs before the search product is created with the goal to encode the information in a way that helps create knowledge out of them. The organisational conducted after the search is a mere wrap-up of gathered materials for future reference. The search product is already created and knowledge has already been acquired.

Conclusion

In conclusion, it can be said that Evans and Chi's model of social search shaped our understanding of search. In particular, the knowledge that search is more than just entering terms and browsing is an important fact of the search process. The *informational search* in their model is the type of search we need to support in our system - knowing that this search is very difficult to conduct, open-ended, iterative and contains the sense-making phase will help us in assessing how to best support knowledge workers in their search.

3.2 Evolution of search


Yahoo! Deutschland Weekly Picks

Search Options

Yellow Pages - People Search - City Maps -- News Headlines - Stock Quotes - Sports Scores

- Arts - - Humanities, Photography, Architecture, ...
- Business and Economy [Xtra!] - - Directory, Investments, Classifieds, ...
- Computers and Internet [Xtra!] - - Internet, WWW, Software, Multimedia, ...
- Education - - Universities, K-12, Courses, ...
- Entertainment [Xtra!] - - TV, Movies, Music, Magazines, ...
- Government - - Politics [Xtra!], Agencies, Law, Military, ...
- Health [Xtra!] - - Medicine, Drugs, Diseases, Fitness, ...
- News [Xtra!] - - World [Xtra!], Daily, Current Events, ...
- Recreation and Sports [Xtra!] - - Sports, Games, Travel, Autos, Outdoors, ...
- Reference - - Libraries, Dictionaries, Phone Numbers, ...
- Regional - - Countries, Regions, U.S. States, ...
- Science - - CS, Biology, Astronomy, Engineering, ...
- Social Science - - Anthropology, Sociology, Economics, ...
- Society and Culture - - People, Environment, Religion, ...

Yahoo! New York - Yahoo! Shop - Yahooligans!

Yahoo! Japan - Yahoo! Internet Life - Yahoo! San Francisco

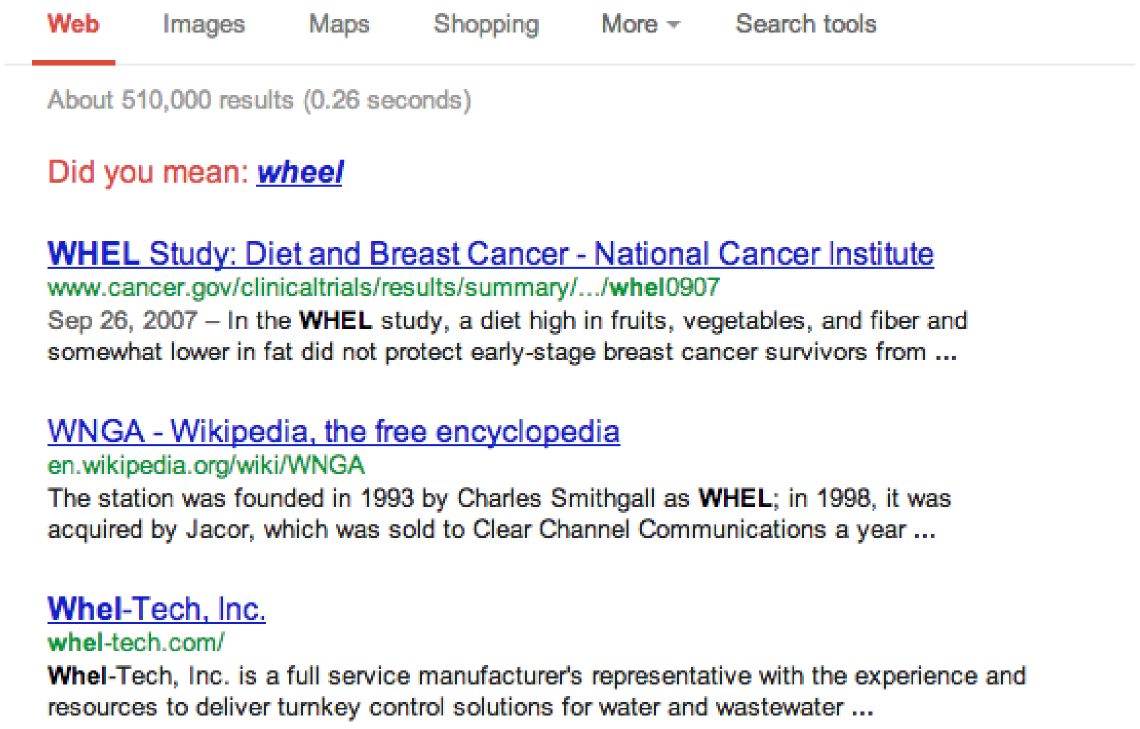
Figure 7: The Yahoo! Homepage in 1996, showing a categorised directory of available webpage. Source: [85]

Now that we know past research in search systems and models of search and therefore have a deeper understanding of the process of search, we will take a look at how search evolved over the last decades. When search engines for local discs, network discs or the web were first introduced, processing power and internet speed were still largely limiting factors. Therefore, the possibilities to search were limited to selected metadata and especially web searches relied on directory-like structures, as it can be seen from the example of the Yahoo! Homepage from 1996 in Figure 7, where site-owners had to manually add their websites to the index. If no fixed, categorised directory structure was used, search results were often cluttered with unrelated documents or spam. Over the years, computers and possibilities to store and retrieve data became much faster and more powerful, allowing a lot of search engines to not only search through a multitude of metadata, but also search the entire content of documents.

Modern examples of such multi-metadata searches are, amongst others, the Google Search[33] for the web and Apple's Spotlight[4] for local discs. At the same time, new and improved algorithms for crawling, indexing, storing, retrieving and ranking results were developed (e.g. [47, 88, 5]). Interestingly, while the technical aspects of search engines improved heavily, certain aspects of search seem nearly unchanged compared to over a decade ago. For example, back in 1998, Zamir et al. already pointed out the disadvantages of unstructured lists as search results:

> Conventional document retrieval systems return long lists of ranked documents that users are forced to sift through to find relevant documents. The majority of today's Web search engines [...] follow this paradigm. [...] The low precision of the Web search engines coupled with the ranked list presentation make it hard for users to find the information they are looking for. Instead of attempting to increase precision [...] we attempt to make search engine results easy to browse. [87]

While the precision of search engines has certainly increased to a certain degree over the years, the presentation as an unstructured ranked list remains. Looking at Figure 8 we see that even today, the world's most famous web search engine, Google Search[32], still uses this approach to present results. While there have been additions and improvements, for example automatic query suggestions ("Did you mean ...") and the possibility to show different kinds of media (Web, Images, Maps, ...), the actual result representation remains largely unchanged.



About 510,000 results (0.26 seconds)

Did you mean: ***wheel***

WHEL Study: Diet and Breast Cancer - National Cancer Institute
www.cancer.gov/clinicaltrials/results/summary/.../**whel**0907
Sep 26, 2007 – In the **WHEL** study, a diet high in fruits, vegetables, and fiber and somewhat lower in fat did not protect early-stage breast cancer survivors from ...

WNGA - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/WNGA
The station was founded in 1993 by Charles Smithgall as **WHEL**; in 1998, it was acquired by Jacor, which was sold to Clear Channel Communications a year ...

Whel-Tech, Inc.
whel-tech.com/
Whel-Tech, Inc. is a full service manufacturer's representative with the experience and resources to deliver turnkey control solutions for water and wastewater ...

Figure 8: Search results presentation by Google Search for the term "whel". It can be seen that Google uses mostly textual lists, but has made some additions in the last couple of years, for example automatic query correction ("Did you mean") and filtering by document type (e.g. images, maps, ...). Source: [34]

In Chapter 2 we discovered the importance of expressing the information need. For example, this problem was already mentioned back in 1992 by Williamson and Shneiderman, who said that formulating a query to express the information need can be very difficult[86]. Of course, back then, they were referring to database systems that required queries in a language like SQL. While there certainly have been improvements in the way users express their information need - users are no longer forced to enter abstract SQL queries, but can use keywords to express their information need and are aided by spelling correction, automatic query expansion or automatic query completion - the problem remains a difficult and important one to solve in modern search systems. Reiterer et al. also pointed out this problem as a major difficulty in search systems[68]. Gerken et al. also mention a good support for this as one of their four basic design principles for VIS systems[25] and Pollock and Hockley carried out two user trials and conclude that "nearly all participants from both trials had difficulty formulating good searching keywords even when they had all the information they needed"[60]. The problem of expressing the information need survived, and almost all major search systems today use keyword searches with the possibility for an advanced search on demand. Obviously, this often remains a technical necessity - especially for web-searches, showing all results is impossible. But this step can not be underestimated and the question arises how to support it best and if there are alternative ways to support query formulation. After all, Borgman points out that a failure to express the information need will often lead to a failure of the entire search process. She says that "users tend to quit immediately after receiving an error message"[12].

In conclusion, we saw that search systems already have come a long way. Besides the increasing speed of computers and networks, there certainly have been improvements like new algorithms to crawl, cluster, rank, index, store, retrieve, multi-metadata searches, additions like query suggestions, query expansion, advanced filtering and more, which help the user in his search. But we also see that there is still a lot that can be done and that certain aspects of search, for example the visualisation of results, still remain largely untouched.

3.3 From information need to search query

The step of expressing the information need, that Evans and Chi call the *requirement refinement* in their model of social search[20, 21], is often the hardest in the entire search process. Gerken et. al list a good support for this step as part of their four basic design principles for visual information seeking systems[25]. Borgman even talked about this problem way back in 1986 and said this is the most error-prone step in search. She said:

> Information retrieval is inherently a complex task. It involves the articulation of an information need, often ambiguous, into precise words and relationships that match the structure of the system (either manual or automated) being searched. [12]

This was backed up by Mußler in 2002[51] and by Borgman herself ten years later in a follow-up paper, where she said:

> Information retrieval is a difficult problem because it requires describing information that you do not yet have. Searchers must translate their information needs into a description of the information sought, relying on their own knowledge of the problem [and] their understanding of the tools the system provides to assist in describing the problem [13]

Every modern knowledge worker knows the situation where zero-hit or many-hit results are returned after querying a search system. Even worse, the situation may occur where the search system interprets the input wrong, returning only irrelevant results. This leaves the knowledge worker with the difficult situation of finding out what went wrong and correcting it, which might require a deeper understanding of the internals of the system. Sometimes, the cause may also be simple - maybe a simple spelling error is the culprit. But oftentimes, the problem is more complex and the user didn't express the information need in a way the system understands it - the entered query was "wrong" from the perspective of the system. In query languages like SQL, the queries are hard to formulate but problems can often be found easily: The compiler will warn the user about unknown columns, wrong data types or other mistakes. But with modern search system, where keyword searches are common, finding these problems automatically is mostly impossible and even for the user its often a long and frustrating process. This is especially true if there is no solution to this problem at all - which can for example be the case with people-searches on the web. As Nuray-Turan et al.[55] pointed out, support for people searches is still fairly limited in modern search engines. Not only the searched person, but all namesakes of that person, show up in the results, leaving it to the user to dig through and figure out the correct results. In a library context where authorships and author relationships play an important role, this can make the difference between a successful and a failed search. Pollock and Hockley talked specifically about the problem of expressing the information need in keyword searches and said that "Nearly all participants [from two trials conducted] had difficulty formulating good searching keywords even when they had all the information they needed"[60]. Grün et al. point out that "Today's retrieval operations should therefore be uncoupled from formal issues, allowing the user a high level of flexibility and freedom"[27].

Additionally, Furnas et al.[22] explored keyword-based searches in general and found several problems with them as well: Above all, they point out the problem of ambiguity of natural language. Even if the system knows multiple references to one term, it is still possible or even likely the user will chose another one unknown to the system. Furnas et al. state that "it is the long-tailed distribution of the word usage [...] that is the villain"[22, p. 47], referring to Zipf's law which states that a long-tailed distribution occurs for word-frequency in natural languages. In other terms, this means that frequent words occur extremely often, but there is a long list of words that do occur, but very rarely. To add to that problem, the exact same term can be used in a search engine describing different information needs and expecting different results. Furnas et al. propose some solutions for increasing the success rate of such keyword-based systems, for example letting the user chose between different guesses of the system on the meaning of each term, but they also point out that the problem of language ambiguity and the translation from natural language to system model remains unsolved.

In previous sections, we already mentioned some possible solutions to these problems. Williamson and Shneiderman introduced *dynamic queries* for the direct manipulation of query parameters[86] and Ahlberg and Shneiderman proposed *dynamic query filtering*[2]. The user should be able to manipulate a query using UI elements and see the results of his manipulation directly. Coupled with *tight UI coupling*[2] this leads to a better understanding of how query parameters affect the search result. We also mentioned methods like *query expansion* and *filtering*, which a lot of modern search system implement to help the user.

We will go into detail on how we support the formulation of the query in Section 4.4 on page 35. For example, we implemented support for *FacetSearch++*, which allows the user to build complex boolean queries without a deeper understanding of boolean logic by using visual query formulation. We also tried to explore new approaches: For example, we included filtering not as an explicit step performed by the user by adjusting search parameters, but rather as an implicit step of the search process that is performed by increasing the degree of interest on individual objects. Thanks to semantic zooming, this will show additional details on the selected objects and other objects will disappear outside of the bounds of the screen. We are confident that with this approach, a natural search flow of work can be achieved that allows to narrow down search results without the explicit use of dedicated UI elements and therefore help the user in searching in a more fluid, natural way.

3.4 Representing information

In our analysis of existing systems in Chapter 2, we also discovered the importance of how information and documents are represented. Having a fast, effective data storage and good algorithms to find relevant information is important, but if the information is not presented in a way the user can - and wants - to explore them, then the information need will not be served. We discovered that a lot of modern search systems rely on an unstructured textual list and good ranking algorithms to present their results to the user, which can for example be seen in Figure 8 on page 25. As pointed out by Drori[18], the two mainly accepted ways to display such lists is displaying the title of the documents, or the title along with the first few lines of the documents. A lot of authors question if this is really the best way to aid the user in finding relevant results in his search, especially in times where the number of results is often overwhelming and it is impossible for users to look through all the returned results. Textual lists do not scale well to large data spaces and large result sets. For example, Büring and Reiterer mention this problem in regard to PDA screens:

> Unfortunately, the majority of interfaces on today's handheld software present information in a textual format, making use of lists and tables, which usually display fewer than 20 entries on a single PDA screen. [14]

While this problem is certainly less severe for desktop screens and screen resolutions increased over the years (although they seemed to have reached a limit in recent years, where DPI increases without the resolution, as for example with Apple's "Retina Displays"), usual list-based results do still rarely fit more than 20 results on a single screen. The world's most famous web search engine, Google Search[32], actually limits the results to ten per page. Also, studies have shown that users usually look at the first few results but no further[31, 81]. This means that, no matter how many results are returned and completely independent of how many and which of them are useful or relevant, most of the result set is lost. If the ranking algorithm performs poorly or the expression of the information need was not good, this problem gets even worse.

When looking at the literature, it seems that little has changed in the last decade when it comes to presenting search results. Even back in 1998, Zamir and Etzioni already mention that the representation of results as lists might be problematic:

> Conventional document retrieval systems return long lists of ranked documents that users are forced to sift through to find relevant documents. [87]

In 2000, Mann and Reiterer explained the importance of visualisations of search results and compared five different visualisations in their INSYDER system[45, 68, 67, 69], one of them being the default list representation found in a lot of search systems. Although they also find severe drawbacks of this representation, it is still the default visualisation of results as of today. Jetter et al. mention the drawbacks of lists, especially in times of multimodal results where the result set can be combined with images, audio or video[36].

So, what alternatives can we use that help the user in efficiently explore the data space, even with large result sets? There has been a lot of research in this area, for example the *WebBook* by Card et al.[15], which displayed webpages in a book-like fashion. They grouped pages of a single domain into a "webbook" and made it possible to explore a domain like one would explore a book and exploring the web like one would explore a bookshelf. Chimera and Shneiderman explored the possibilities of table-of-content-structures[16]. With the "Superbook", Egan et al.[19] researched, amongst other features, the advantages of hypertexts - digital documents that allow nonlinear exploration by hyperlinking parts of the texts. Zamir and Etzioni explored the possibilities of document clustering[87]. They mention the Suffix-Tree clustering, a clustering method that groups documents by common phrases. A somewhat similar approach is taken by Rauber and Merkl[63], who proposed a way to organise large document collections and structure them in a way that helps the user in exploring large data spaces. They used self-organising maps (SOMs), a form of neural network, to cluster similar documents and then present them to the user. With neural networks, though, labelling clusters is difficult - the neural network is basically a black box. They solved this problem with a custom labelling method called *LabelSOM*[62]. When looking at those solutions, we certainly saw major advantages in the clustering approaches - they give the user an abstract view on the data space, shrinking down large result sets and making them explorable. They allow to display more results than it would be possible with simple lists.

In conclusion it can be said that textual lists are probably not the best possible representation of documents. They were developed decades ago and are nearly unchanged since then. They don't make much use of modern possibilities or technologies like high-resolution screens, fast hardware, multi-devices-usage or newly developed ways of interaction like multitouch. There has been a lot of research of how to alternatively display individual documents or large document spaces, but little of them have yet had success in the general public or commercial systems. In particular, clustering algorithms seem like an interesting and promising approach. We will propose our solution to this problem in Chapter 4 on page 31.

3.5 Sense-making

In Section 3.1.3 on page 22 we talked about the sense-making step and its part in the Canonical Model of Social Search by Evans and Chi[20, 21]. While there are some minor differences between the definitions of sense-making of different authors (for example, Evans and Chi separate foraging and sense-making as two different phases), they basically agree. Pirolli and Card put sense-making tasks in a nutshell:

Information → Schema → Insight → Product [59, p. 2]

This means that, when a knowledge worker takes on a task where sense-making is part of the process, there are several successive steps involved:

1. **Gathering information**: The information need must be defined and expressed ("requirement refinement"[20]). Then, information is collected by looking at documents in the search result and relevant documents are bookmarked or somehow marked for further exploration later. The gathered files ("evidence files"[20]) are then skimmed or read in depth with the goal to either discard them or to extract information from them.

2. **Encoding**: The gathered information are re-encoded in a schema - the representation of the information shifts. The goal here is "a more compact representation of the essence of the information relative to the intended task"[59]. Pirolli and Card call this the "representation shift loop"[59]. This can for example mean simply copying relevant passages of the documents to a word file, or creating a mind-map or putting keywords on cue cards etc.

3. **Gaining insight**: The new schema allows the knowledge worker to gain insight from the information. For example, encoding the information in a mind-map could lead to a very compact view on the information that allows the user to see connections he did not see earlier. This insight, or knowledge, is the goal of the search process itself and is what satisfies the information need.

4. **Creating a product**: From the gained insight a final product is created. Evans and Chi illustrated this well in their model of social search with the “After Search”-phase, where the knowledge worker creates a search product, for example a presentation, an essay etc.

It must be mentioned that this process is highly iterative. Gaining insight can lead to new searches or creating the final product can reveal missing information that in turn makes need for a new search.

In conclusion, sense-making is often described as an integral part of the search process. Allowing the user to express his information need and representing search results adequately is important, but eventually the user has to create knowledge out of the results. Often, the sense-making is distributed over a lot of applications because the search systems do not support it. Following the example of *MedioVis 2.0* as a *Knowledge Media Workbench*[29] we want to support a knowledge worker’s workflow in an integrated fashion. This way, the user can stay within the context of one system during his work. As part of this approach, we will integrate support for the sense-making phase, which we will lay out in detail in Section 4.6 on page 55.

CHAPTER 4

SEARCHING IN THE BLENDED LIBRARY

4.1 The Blended Library

At the group of Human-Computer-Interaction at the University of Konstanz, the *Blended Library* project[6] integrates traditional library workflows with state-of-the-art computer-aided interaction systems. The BL frequently hosts new projects and the setup is changed and enhanced constantly, but the current setup is similar to the one shown in Figure 9.

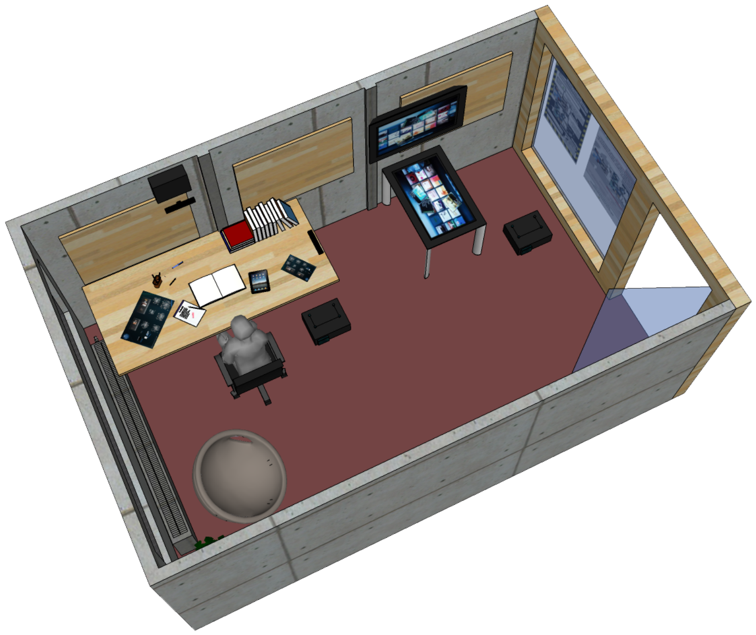


Figure 9: The Living Lab of the *Blended Library* project as a 3D model. Source: [71]

The desk (top left in Figure 9) allows workers to perform traditional library work, for example reading books, magazines or digital media, taking down notes with pen & paper and work on their personal digital devices (tablets, laptops, ...) to conduct research on the internet, create presentations and so on. Going further, the desk allows the user to take down notes and annotations on available Anoto paper[1], digitalising the written notes instantly and making them available on digital devices.

Additionally, a projector with an attached camera has been put above the table and the table has been enhanced with a multitouch-aware coating. This allows the worker, but also projects in the *Blended Library* to project digital media onto the table that can then be interacted with by means of multi-touch gestures. For example, documents, websites or annotations could be displayed, created, altered and moved on the same table that the user is working on with real-world materials. A system to detect so-called "blobs", for example books or arms, prevents the false detection of real-world objects as touch interaction.

In the lower left there is a comfortable reading chair that the knowledge worker can use for in-depth reading sessions. Besides traditional books and magazines, modern and lightweight tablet computers can be used to read digital documents in this area. When using digital media, the worker can also be aided in his reading, for example by using interactive reading systems[10, 83, 82]. The goal is to allow the worker to get a deeper understanding of individual documents or simply to relax from his work.

Opposite of the desk, a board on the wall (not depicted in Figure 9) allows multiple people to work collaboratively on a single task, for example when brainstorming or presenting results of their work to others. Additionally, installed projectors allow to show digital media on the board and therefore seamlessly merge real-world drawings and text with digital information. The projectors are installed in a way that they project from above the board, which allows people to stand in front of it without their shadow obscuring the projection.

Further digital support is available at touch displays, namely two Samsung SUR40 with Microsoft PixelSense[49] devices. These 1080p, 40" screens with optional table legs are equipped with the Microsoft PixelSense technology, making them multi-touch devices. One of the SUR40s is used as a touch table, the other is mounted to the wall and can be rotated freely. This allows the wall-mounted device to be used in landscape or portrait mode, depending on the need of the users or the need of the application. The tables can be used to display large amounts of information, advanced digital visualisations, do research, search the local library content and more. Also, the size of the devices allows for collaborative work to take place here. The table devices additionally can detect so-called *tokens* - real-world objects that can trigger actions in an application if placed, removed or moved around on the table. For example, an application could detect a mobile device and send data to it when it is placed on the table or detect a book and display metadata and annotations.

4.2 We in the Blended Library

As mentioned in Section 1.1, we discovered a lack of support for the steps of exploring and evaluating search results and making sense of documents in the BL. For this reason, we created an application prototype that supports the search & sense-making process that was discussed in detail in the previous chapters. One of the early decisions in the project was to make use of the advantages of the ZOIL paradigm and ZOIL framework[89]. The ZOIL framework is a multitouch-based framework that allows the application to put objects on an infinite, two-dimensional landscape and zoom and pan that landscape freely. ZOIL also introduces the concept of semantic zooming, which is very helpful in giving the user the possibility to view different dimensions of the data space while staying in a single context. In addition, we decided very early on that we wanted to support *FacetSearch++* as a way of expressing the information need and formulating a query. Since *FacetSearch++* is built for the Samsung SUR40 with Microsoft PixelSense, the prototype should allow the user to stay in the same multi-touch based interaction context. Switching between a multi-touch device as the means to formulate the query and, for example, an ordinary desktop system to browse the results, would require constant context switching and be an adverse and non-desirable solution. Because of these circumstances we decided to implement our prototype with multi-touch support using C#/WPF. For a long time, the system was developed for and on the Samsung SUR40. When the new touch-enabled desk was added to the Blended Library, that desk was adopted as the new platform for the application. Besides the fact that the system could be ported with almost no additional effort, the desk allows the user to conduct searches at the same place where he performs his real-world library work - the need to actively switch to the Samsung SUR40 disappeared. Additionally, the table brought support for pen-interaction. Since supporting sense-making would be part of the prototype, and distinguishing between normal and annotational interaction had been a problem so far in the development, this proved to be a great addition. It also gave the user the possibility to use a touch-pen to draw free-ink annotations - a great benefit for the sense-making step.

In the following sections we will discuss our prototype, its concepts and design decisions and how we implemented them. We will first give a broad overview over the system architecture in Section 4.3. We will then go through the different stages of using our prototype, beginning with how to start a search and express the information need in Section 4.4 on page 35, how to explore and browse the search results in Section 4.5 on page 39 and how we support sense-making in Section 4.6 on page 55. We will conclude our prototype implementation in Section 4.7 on page 57.

4.3 System Architecture

4.3.1 Hardware

The previous section dealt with the technical setup. Most of the development of the system was done on the Samsung SUR40 with Microsoft PixelSense[49]. The display of the Samsung SUR40 is a 40" touch table with a 1080p resolution. The table itself features an Athlon X2 Dual-Core 245e processor with 2.9GHz per core, an AMD HD6750M GPU, a Realtek ALC262 audio card, an SATA2 hard drive, 4GB of DDR3 RAM and HDMI, USB and ethernet ports as well as Wi-Fi. The installed operating system was Windows 7 with touch support. Visual Studio 2010 with the Surface SDK and Expression Blend were used as the development environment.

In the course of his master's project[24], Christoph Gebhard, with support by Roman Rädle, set up a new touch table device in the *Blended Library*. They converted the ordinary wooden table that was available in the BL to a touch sensitive table. For this, a 1080p projector was connected to a computer and projected its image onto the table. This allowed the table to be used as a second monitor. With that setup, the DISPLAX SKIN MULTITOUCH[17], a special coating that allows for capacitive detection of touches, was applied onto the table. Sensors pick up touches on the coating and transfer the data over USB to the same computer that is connected to the projector. The computer runs the *DISPLAX SKIN multi-touch control panel*, a client application that allows for configuration of the touch coating and detection of touches. The application can sent the touches directly to the operating system and enable touch, but this feature was not used. Instead, a custom application called the *WPFEventBridge* was developed. It receives touch data from the DISPLAX client over the TCP/IP protocol and implements custom handling of those events. For example, the *WPFEventBridge* allows an application to distinguish between touch events with the finger and events sent by a pen, which will be received as mouse events. In contrast to the DISPLAX client, which works on an operating system level, the *WPFEventBridge* works on an application level - a WPF application has to include the bridge to enable touches. Closing the application will turn off the touch support. The table was connected to a computer running Windows 8. Visual Studio 2012 and the Surface SDK were used for development on that machine. Switching to this new technology in the course of the project gave us several advantages, one certainly being the integrated fashion of table-based research with real-world documents or personal devices like laptops and searching with our prototype. Another was the advanced event detection that allowed us to include pen interaction and clearly distinguish it from ordinary touch interaction.

4.3.2 ZOIL framework

The decision to use the ZOIL framework was made very early in the development process. The advantages of the ZOIL framework are apparent, as it implements a lot of the important aspect of VIS systems that we discovered in Chapter 2. For example, it makes heavy use of the concept of direct manipulation interfaces as they were proposed by Williamson and Shneiderman[86]. It is also very close to the vision of *ZoomWorld*[61] and adheres the four principles of VISS by Gerken et al.[25]. The ZOIL framework allows an application to create two-dimensional landscapes and to put arbitrary objects on the landscapes that are visually represented and have a spatial relationship to each other. The objects can be moved, rotated, resized and zoomed into freely. The support for semantic zoom allows objects to change their appearance based on the space they have available. This gives the user the possibility to view different dimensions of objects while staying in the context of the information landscape. It further allows the user to work on the entire search process inside a single context, without the need to switch applications or even application windows or tabs. It also supports modern multi-touch interaction to manipulate the landscape.

4.3.3 Programming environment

The hardware decisions and the decision to use the ZOIL framework as the foundation of our concept defined the language and architecture the prototype was built on. It was programmed in C# and built on Microsoft's WPF API. The WPF API gives us the necessary tools to build a multi-touch ready system that can detect common gestures like tapping, pinching and panning and system events like keystrokes. It also provides us with a number of predefined, multi-touch optimised UI elements like buttons, text boxes, scroll views and more. These elements already come with the necessary event handling for detecting touch events and were styled and sized in a way that made them easily usable with fingers. Additionally, as they are provided by Microsoft, their general style fits that of ordinary Microsoft Windows touch applications and therefore complies with user expectations.

4.3.4 Portability

An aspect of future expansion possibilities was the portability of the system. We programmed the application in a resolution- and device-independent way. The only requirement was that it was a multitouch-enabled device running Microsoft Windows. This was an advantage when we ported the application from the Samsung SUR40 to the custom table in the Blended Library. But this concept can be expanded further - for example, the application could be ported to a mobile device and used on-the-go while walking through the library or on the users personal mobile device in the future.

4.3.5 Data Source

Besides our prototype system we connected only to a single external source - a BaseX[26] database that provided us with information about the documents we displayed. It contained 7305 publications. The database was pre-existent and fulfilled the requirements we needed: It was easy to use, fast, allowed for parallel data access of multiple users and a C# driver was readily available. Because of this, we saw no reasons to transfer the available data to another database. Each document in the database contained a multitude of metadata including the title, authors, year, language, number of pages and more. BaseX is an XML-based database and objects are returned as XML, but the C# database driver converted them to C# objects. A detailed example of the XML of a single document can be found in Appendix A on page 77. It should be especially noted that all of our documents had an abstract attached. A lot of them also had the full text attached, but this was not guaranteed.

The data arrives in the application via BaseX database events. When a token is placed or removed from *FacetSearch++*, a *ShowResults* or *RemoveResults* event is sent that includes the query at the tokens' position and the ID of the token. From that query the documents to add or remove can be retrieved from the database. The returned C# objects were then processed to store additional metadata like the query and token that belong to the object.

It is important to note that the connection to the database was built completely modular. It was important to us to build the system with expansion in mind. A new database, for example results from Microsoft's Academic Search[50], could be build into the application by creating a module for it.

4.3.6 Visualisations

When it comes to visualising the data space, we also built the application with expansion in mind. Although we implemented a single workflow in the final prototype, we built the visualisations modular so they can be easily extended or replaced based on arbitrary parameters, like the kind of objects we are displaying. For example, a simple scatterplot visualisation, showing the documents by year and popularity, could be used if there are less than fifty results. For more results, are more advanced visualisation could be used. Additionally, the user could switch visualisations with a simple button press. The only requirement is that visualisations have to implement a common base class. In our prototype this modularity is currently only used for simple scenarios, for example, to display a text instead of a landscape with objects when no result could be found for a search. In the future, this could be extended to make it possible to display a completely different visualisation for results or to make use of portals similar to the ones used in MedioVis 2.0[29].

4.4 Starting a search

After the *requirement refinement* the user approaches the search system with an idea of what to look for and needs to make the system understand that idea. A lot of modern search engines use keyword-searches to do that, which have their own advantages and pitfalls. Our prototype tried to ease this step as much as possible, and we finally implemented two different ways to search:

4.4.1 FacetSearch++

As the first approach, we implemented support for *FacetSearch++*, a project that was created as part of the *Blended Library* (as discussed in Section 2.13 on page 17). *FacetSearch++* uses the concept of *FacetStreams*[9] to allow the user the visual creation of complex boolean queries without a deeper understanding of boolean logic. This can be extremely useful, considering that Shneiderman et al. describe the decision about boolean operators as one of the most difficult ones in the query formulation step[78]. The idea behind *FacetSearch++* is that the entire available data space (the database) is represented by a stream. Users can put colourless tokens inside that stream and define a filter on that token. The stream(s) that pass through that token are then filtered by the defined criteria. For example, a token could define a year filter and allow only documents from 1980 or earlier - when a stream passes this token, the stream that comes out the other end will represent only documents of that time era. Users can put as many tokens as they want onto the device and connect them to perform multiple filter steps. Streams can also be branched and merged which allows the implicit creation of boolean logic and it also allows users to take different approaches to a search and see the outcome of each branch. A video that shows the *FacetSearch++* concept in detail can be viewed at [7].

The user then has the possibility to put coloured tokens at any position inside the streams. The query that the stream represents at that location will then be sent to our application prototype. There is support for adding up to three coloured tokens and the results of the different searches will be merged in our prototype. This allows the user to try out different search strategies and then combine the ones that are interesting. We additionally implemented features that take special advantage of this multi-query approach, which we will outline in Section 4.5.4 on page 49.

The decision to use *FacetSearch++* as the query formulation step brought several advantages: As we discussed, finding a suitable representation of the information need is difficult, error-prone and not fully solved with keyword-searches to date. *FacetSearch++* solves this problem by incorporating what would be an "advanced search" in most systems in a very easy-to-use and playful way. Putting a token on the application brings up a number of possible metadata. Selecting one of them brings up a selection of meaningful and suitable values to filter by, depending on the type of the selected data. This allows the user to filter the data space step by step. Putting additional tokens on the screen allows for additional filters to be applied. Additionally, the visual representation as streams to represent the data comes very naturally to users, as an evaluation of *FacetStreams* showed[37]. Combining different filters to form even very complex boolean queries is much easier to users as it would be by, for example, typing parenthesis and words like "OR" or "AND" in an input field.

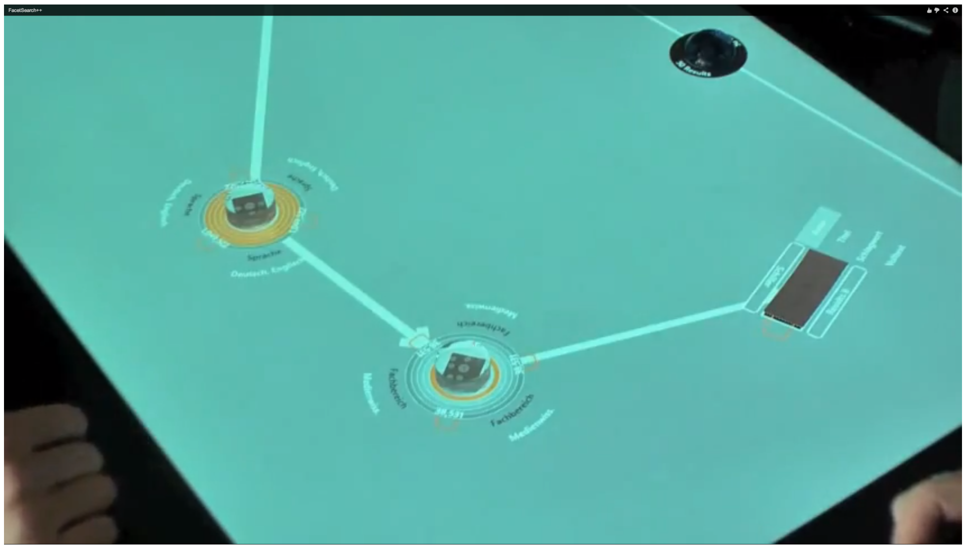


Figure 10: The *FacetSearch++* concept, allowing users to build boolean queries using tokens connected via streams. At the top right is a token inside a stream, sending data to our prototype application. Source: Extracted from the video at [7]

We are confident that we will be able to ease the step of expressing the information need significantly by incorporating *FacetSearch++*. By giving the user a defined point of entry to our system - putting a token on the application surface - we lessen the difficulties in getting started with searching and the concept of streams makes it easier for the user to form complex queries.

4.4.2 Entrance Search

The prototype was supposed to offer an integrated solution of the search process and *FacetSearch++* might not always be available, for example when our system might be used on a mobile device. Because of this we also pursued an alternative approach that didn't rely on an external application. The VIS system therefore features an *all-in-one-search*, inspired by the one proposed by Grün et al. in *MedioVis*[27]. It appears as the first screen of the application in case *FacetSearch++* is not available, as seen in Figure 11. This search differs from an ordinary keyword search in that it is truly multi-metadata - searching across title, content, authors, year of release and more from the keywords entered in a single input box. For the design of the input box, we pursued the suggestions made by Shneiderman et al.:

> One additional rule specific to text-search interfaces is worth mentioning [...]: Allow plenty of space in text-entry boxes. This is particularly important because longer search text very often gives better recall and/or precision, and so users should be encouraged to use long search strings. [78]

The user is greeted with a single, large text-entry field that additionally dynamically enlarges itself when a lot of text is entered. Text is never cut off. This way the user is encouraged to use long search terms and not be shy about the length of the search query.

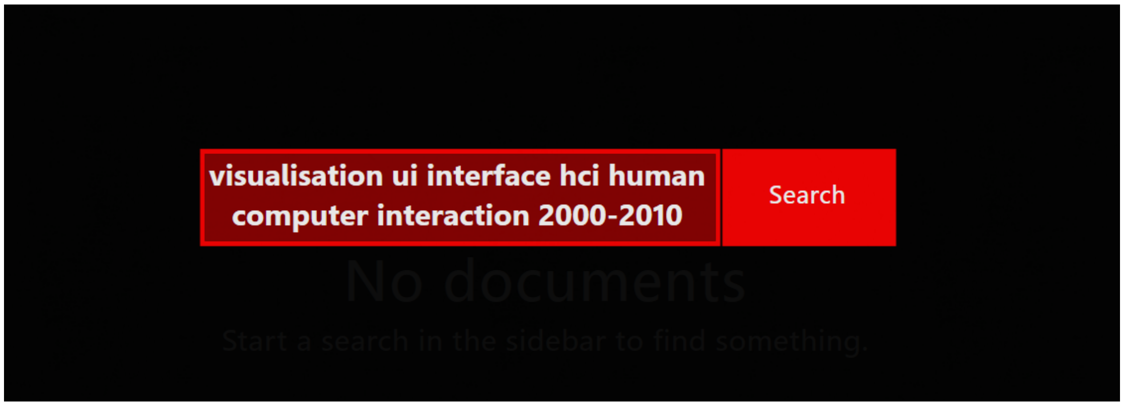

visualisation ui interface hci human computer interaction 2000-2010
Search
No documents
Start a search in the sidebar to find something.

Figure 11: The start screen of the application in case *FacetSearch++* is not available. It features an *all-in-one-search* that allows for a smart, data detecting cross-multidata search and additionally allows to search the data space by topics based on a topic-modelling.

Furthermore, we implemented different data detections. They intelligently detect the type of data the user wants to search through and build the query based on that. For example, the user might search for the string "shneiderman 1990-2000". The system would search for "shneiderman" across textual fields like author, abstract, etc. And it would detect that the second term is a range of years and filter the result set by publications that appeared between the year 1990 and 2000. These data detections allow the user to use a natural approach to keyword search while eliminating the need for additional textual operators like "year:". With this approach we follow in the footsteps of *MedioVis*, where Grün et al. said that "Today's retrieval operations should [...] be uncoupled from formal issues, allowing the user a high level of flexibility and freedom"[27].

On top of that, we added a topic-based search. In addition to detecting the type of data the user wants to search, this makes it vastly different from the searches that we found in other research. We perform a topic-modelling on the documents of the data space (details on that in Section 4.5.1 on page 40) and the all-in-one-search allows the user to search through the results of that topic-modelling.

In other words, this allows a user to search for documents that belong to a specific topic, resulting in documents from that area of expertise. For example the search "visualisation 2000-2010" will bring up documents released between 2000 and 2010 that in some way revolve around the topic of visualisation. In our opinion, this will allow the user to express his information need much more straightforward, as it allows to map the mental model very directly to a search query. Instead of having to do the difficult transformation from information need ("I want to know about topic X") to search query ("keyword1 keyword2 keyword3"), the user can now search for what he is actually looking for.

Literature taught us that it is important to support searching as an iterative process, where each iteration further refines the search. Besides others, Shneiderman talked about the iterative nature of the search process in his Framework for Mega-Creativity[77]; Evans and Chi's Canonical Model of Social Search points out the iterative nature of the *informational searches* as a central point[20, 21] and Pirolli and Card also stress that the search process is highly iterative[59]. Additionally, Ahlberg and Shneiderman define the refinement of the query as a major difference between browsing and ordinary information retrieval[2].

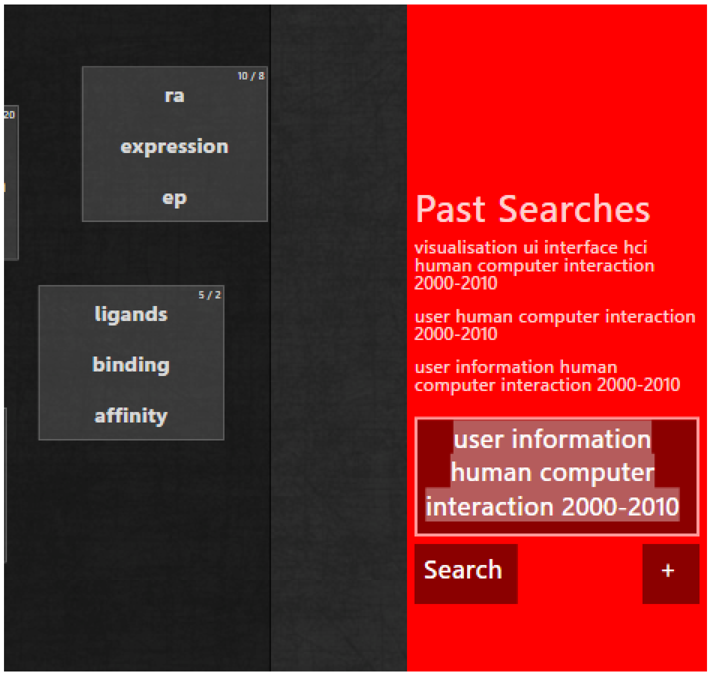

10 / 8
ra
expression
ep
5 / 2
ligands
binding
affinity
Past Searches
visualisation ui interface hci human computer interaction 2000-2010
user human computer interaction 2000-2010
user information human computer interaction 2000-2010
user information human computer interaction 2000-2010
Search
+

Figure 12: The search sidebar. It can be called at any time with a simple swipe gesture. It features the possibility to refine the current search, start a completely new search, see and search for past searches and use the *tap to search* feature to add text from the application into the search query without typing.

To support these requirements, we implemented the so-called *Search Sidebar* to our system that can be seen in Figure 12. This sidebar is available globally in the application by swiping with the finger from the right edge of the application with a single finger. This will slide the sidebar in from the right. The sidebar contains different features (explained further below), with the most important ones being put at the bottom of the bar. This is different from ordinary UI design, as the application is designed for use at a touch table. When sitting at such a table, the UI elements at the bottom (nearest to the user) are the ones that can be reached the easiest. The sidebar features a single-input search textfield, similar to the one the user is greeted with when first starting the application (Figure 11). This textfield is also designed large and automatically expands in size when the user enters long search queries. Search is started explicitly when tapping the "Search" button.

We additionally implemented a feature called *tap to search*. Because typing on multi-touch devices, especially on mobile devices, is not always pleasurable and sometimes difficult for novice users, the user can press and hold the "+" button of the search sidebar to active *tap to search*. This will transform the sidebar to an active state (changing its background colour, see Figure 13) and the user is then able to tap a text in the application (Figure 14) and that text will be added as a query term to the current search query (Figure 15). For example, the user can tap an author or a word in a topic's word cloud and that name or term will be added to the current search query. With this addition to the search, we make it easier for the user to formulate his query, even for novice users and under difficult circumstances.

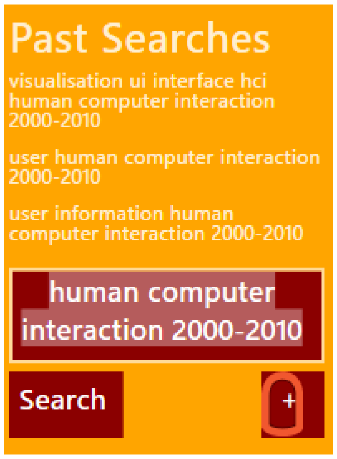

Past Searches
visualisation ui interface hci human computer interaction 2000-2010
user human computer interaction 2000-2010
user information human computer interaction 2000-2010
human computer interaction 2000-2010
Search
+

Figure 13: *Tap to search* can be activated by holding the "+" button on the search sidebar. This will activate tap to search, marked by a different background colour as long as the "+" button is held.

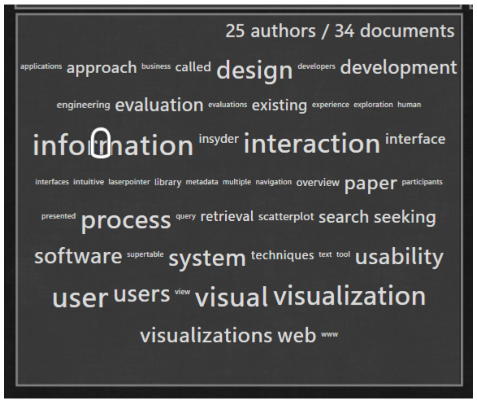

25 authors / 34 documents
applications approach business called design developers development
engineering evaluation evaluations existing experience exploration human
information insyder interaction interface
interfaces intuitive laserpointer library metadata multiple navigation overview paper participants
presented process query retrieval scatterplot search seeking
software supertable system techniques text tool usability
user users view visual visualization
visualizations web www

Figure 14: While *tap to search* is active, a word in the application can be tapped in order to add it to the search query. In this case, "information" is tapped.

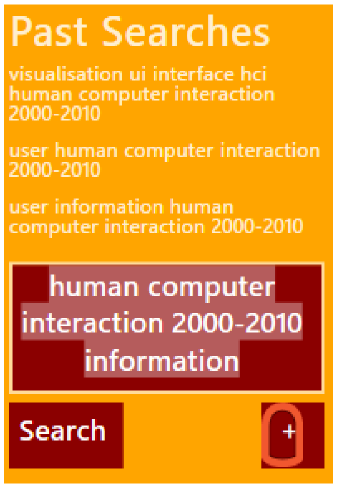

Past Searches
visualisation ui interface hci human computer interaction 2000-2010
user human computer interaction 2000-2010
user information human computer interaction 2000-2010
human computer interaction 2000-2010 information
Search
+

Figure 15: The result of the tap in Figure 14. "information" is now part of the search query. Releasing the "+" button will return the search sidebar to its normal state and the new search query can then be executed.

In his four-phase framework for textual search, Shneiderman pointed out the usefulness of search history to trace back the steps taken during a search process[78]. Because of this, we implemented a history feature into the search sidebar. Past searches can be reviewed in the sidebar and tapping a past search will put that search query in the search textfield - the action to start the search is left as an explicit action to the user, though. This was done to give the user the opportunity to refine the search and to prevent current search results from being removed accidentally. With this feature, past searches can be traced back and reevaluated.

4.5 Presenting the search results

When a search is executed, the next thing a user expects to see are the results. We have seen in Section 3.4 on page 28 that a lot of modern search engines still fall back to a textual list with metadata filtering when it comes to presenting search results. We also talked about the problems of this approach and about possible solutions, of which we implement some in this section and also propose some new approaches.

One of the main goals we wanted to achieve was a natural flow of exploration. In common search systems, especially browser-based web searches, the user has to switch context constantly, which requires a high cognitive effort. The results open in a browser window, adjustment of the search filters is performed on another page, opening each of the results to explore them is done in a different browser tab and so on. Following Gerken et al.'s design principle of providing "views to different dimensions of an information space"[25] and the multi-dimensional view approach of systems like INSYDER[68] or VisMeB[44], users must be able to move seamlessly between dimensions without a constant change of context. Another important and obvious goal was also to use visualisations that make it easy to find relevant results as quickly as possible. Futhermore, in a time where most students are used to working with touch screens and multi-touch gestures, using the ZOIL framework and C#/WPF utilises known, tried-and-true interaction techniques. The user will feel right at home if he has worked with a multi-touch device before, while novice-users will still be able to learn the interaction techniques quickly. In the following, we will talk about the different stages of the search result visualisation. We will explain how we visualised the result set and why we did so and how to move between different dimensions of the results.

After the search query was sent to the server, the user is presented with a first visualisation that is already very different from traditional, list-based results:

4.5.1 Topic-based search results

Visualising large result sets is very challenging. We want to give the possibility to explore as much of the result set as possible without overwhelming the user. Furthermore, users only have limited time to explore a data space - if the user has to look through all the documents of a result set, he will have to stop after only a fragment of the results and information will be missed. Therefore, finding a way to make it possible for the user to quickly separate relevant parts of the data space from irrelevant parts is crucial.

We saw clustering the data space as a possible solution in Section 3.4 on page 28. This seems like a good approach, since it allows us to break down a large data space into a more abstracted view. This makes it possible for the user to get an overview, even over large data spaces, before looking at details of selected parts of the data space. The selection of those parts and the switch to a more detailed view can be implemented using the semantic zoom capabilities of the ZOIL framework, similar to the movies grouped by genre in *MedioVis 2.0*[29]. Two clustering-based approaches we saw were by Rauber and Merkl[63, 62] and Zamir and Etzioni[87]. Rauber and Merkl proposed their system *SOMLib*. It uses a tf-idf[80] based self-organising map, a form of neural network, to cluster documents by similar content and present those clusters instead of each individual document. This retains information about the documents in each cluster, because the clustering groups together documents with similar content and their additional, *LabelSOM*[62] allows to create labels from those similarities. A similar approach was taken by Zamir and Etzioni, who used suffix-tree clustering that clusters documents by their most common phrases.

As we will mention in Section 6.1 on page 66, comparing the different clustering algorithms with regard to computational complexity and applicability for our system would be important in developing the prototype further. In this bachelor project, though, a long comparison and implementation phase of the clustering algorithms was out of scope. This is why we did pick up the idea of clustering documents by content, and used a linearly-performing algorithm that was fast to implement and performed well to proof our concept of a topic-based approach of presenting search results. We used the *latent dirichlet allocation* (LDA)[11], a popular topic-modelling algorithm that is based on tf-idf. Similar to the SOMs by Rauber and Merkl, the LDA clusters documents by similar content. Semantically, it works differently as it is specifically used for topic-modelling. The LDA determines groups of words that often occur together in the same document. Each of those groups is considered a topic. Documents are then assigned to those topics. This has the additional benefit of implicitly generating labels for the topics (the words in the group), which is important for us as it allows us to display those labels to the user to make the user understand the contents of a cluster.

The results of such a LDA-based topic-modelling can then be seen in Figure 16 and Figure 17 . We will now go into more detail on how the LDA-based topic-modelling works, how we further processed the modelled topics and then about an additional clustering step we performed, where we additionally clustered the topics.

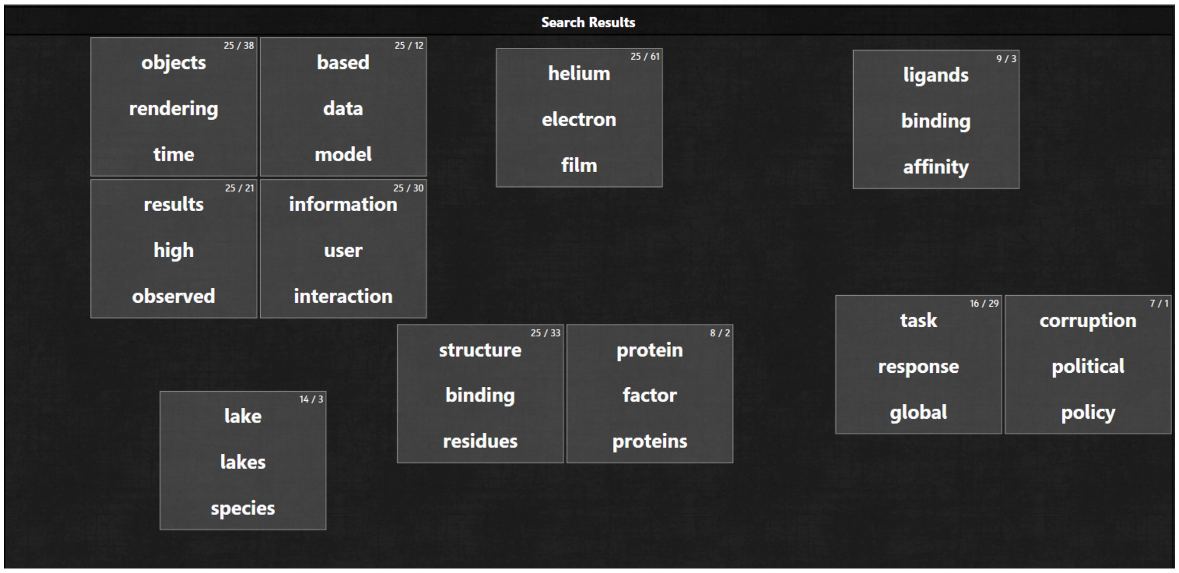


Figure 16: A search result as it is presented to the user. Documents in the result are assigned to topics via an LDA topic-modelling. Instead of presenting every single document to the user, a more abstract view is given by presenting topics with their most important words, and further details available on-demand via ZOIL's semantic zooming.

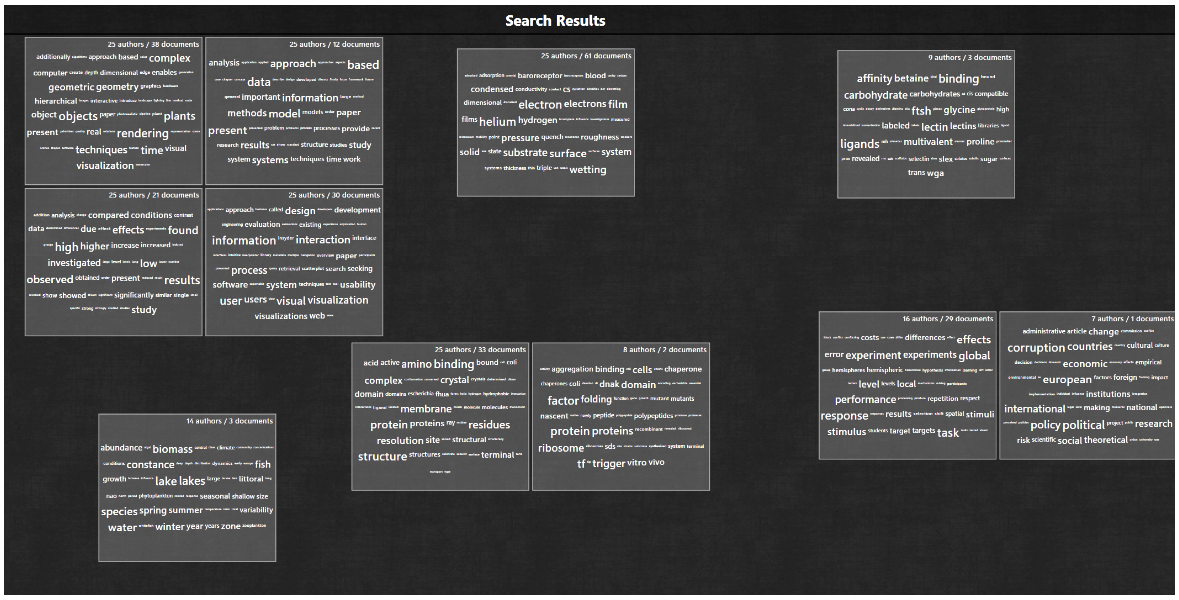

Search Results
25 authors / 38 documents
25 authors / 12 documents
25 authors / 61 documents
9 authors / 3 documents
25 authors / 21 documents
25 authors / 30 documents
16 authors / 29 documents
7 authors / 1 documents
25 authors / 33 documents
8 authors / 2 documents
14 authors / 3 documents

Figure 17: The same search results as in Figure 16. Because of the semantic zooming capabilities, the topics reveal more of their content after zooming in further. Here, instead of displaying only the three most important words, a word cloud with the fifty most important words is displayed. Font size determines the importance of a word for the topic.

Topic-modelling The LDA algorithm determines topics of documents in the document space and then assigns each document with a certain likelihood to each topic. Topics consist of words, ranked by importance inside that topic. The number of topics is given to the LDA as a parameter.

We used MALLET's implementation of the LDA algorithm[46]. The topics were created on the document abstracts because full texts where not available for all the documents. Two things are calculated by the algorithm:

- Topics: The algorithm calculates a fixed number of topics and each topic consists of a dynamic number of ranked words. For example, a topic could be made out of the words 'interaction, user, interface, usability, human, psychology' in that order, which would mean that 'interaction' is the most important (most often occuring) word, then 'user' and so on. The LDA algorithm does not assign labels to topics, topics are defined only by the words they consist of - but a human is still able to conclude that the above topic is, for example, probably related to human-computer interaction. It is important to mention that topics created by the LDA can overlap - this means that words can occur in more than one topic.
- Document-topic affiliation: The algorithm calculates the likelihood that each document belongs to each topic, whereas the probabilities for a single document add up to exactly 1.0. In other words, imagine a pie chart being built for each document, whereas the individual slices of the chart represent topics. If the words of a topic occur often in a document, the topic's slice gets bigger and the affiliation is higher. But this also means that other topic's slices will get smaller, since every document must end up with exactly one full pie.

The algorithm returns the results of the LDA as a simple text file. When we use the input parameters `n=number of topics` a text file in the following format will be returned:

```
Document {ID} {PROB 1} {PROB 2} {PROB 3} ... {PROB n}
Document {ID} {PROB 1} {PROB 2} {PROB 3} ... {PROB n}
...
Document {ID} {PROB 1} {PROB 2} {PROB 3} ... {PROB n}
TOPIC: {WORD 1}, {WORD 2}, {WORD 3} ... {WORD k}
TOPIC: {WORD 1}, {WORD 2}, {WORD 3} ... {WORD i}
...
TOPIC: {WORD 1}, {WORD 2}, {WORD 3} ... {WORD j}
```

Listing 1: An example of the format of a text-file as it is returned by the LDA.

First, a list of all input documents is returned. Each `ID` is the document ID from the database (passed as input). Each `PROB` is a document-topic-probability. `PROB 2` of the document with `ID=10` gives us the probability that document 10 is related to the second topic returned by the LDA. This also means that each line beginning with `Document` represents exactly one document and that each of these lines also contains exactly n probabilities, one for each topic.

Next, the generated topics are returned in the file. A bag of words defines each topic. The words are ranked, so the first word is more important than the second one and so on, but the actual importance is not given. The order of the topics does not have any meaning, but they are ordered in the same way that the probabilities are in the document lines. So `PROB 1` of each document refers to the first topic in the text file and so on.

To use the generated topics in our application, a dedicated parser was written that was able to parse such an LDA-generated text file and transform it into `Topic` and `DocumentTopics` objects. The first represents a single topic and contains all words of that topic, the latter one defines a document and contains a reference to all `Topic`-objects the document belongs to and their respective probabilities. This way, it was easy to access all topics for a certain document and vice versa, the most important topics for a document, the topics with the most documents and so on.

Clustering documents into topics After the topic-modelling is complete, we continue to preprocess the documents in the result set to prepare them for the visualisation. Because a topic can contain many documents, the user is presented with another layer of abstraction to help him get an overview over an individual topic. Authors are displayed when the user explores a topic, and only when the user explore individual authors, documents of those authors are displayed. To achieve this, we group documents by authors and each group contains the documents written by the author. The application then works with those groups instead of individual documents. This is done to further help the user in getting an overview over large result sets.

We will now present the pseudo-code of the algorithm that determines which authors are assigned to which topics and which topics show up on the screen at all. After that, we will explain the pseudo-code in more detail.

Listing 2 Algorithm that groups documents in the search result set by their authors and assigns those authors to topics determined by the LDA. It then determines which authors and topics are displayed.

```
Input: The list of documents in our result set
Output: Topics with assigned authors on the screen

authors ← documents grouped by authors

// Calculate topic probabilities for each author by averaging the documents
for each author in authors do
    topicProbs ← new array
    for each document in author do
        for each topic of document do
            topicProbs[topic] ← topicProbs[topic] + topic-document probability
        end for
    end for
    finalProbs ← new array
    for each key in topicProbs do
        finalProbs[key] ← topicProbs[key] / number of documents of author
    end for
    save finalProbs to author
end for

// Assign each author to one or more topics
// MultiValueDictionary is a custom dictionary with multiple values per key
topicsWithAuthors ← new MultiValueDictionary
for each author in authors do
    authorTopics ← new array
    for each topic of the author do                    // calculated and saved above
        if topicProb >= MIN_AFFILIATION then
            add topic to authorTopics
        end if
    end for
    sort authorTopics by probability
    relevantTopics ← the first MAX_TOPICS topics in authorTopics
```

Listing 2 (continued)

```
    for each topic in relevantTopics do
        add author to topicsWithAuthors[topic]
    end for
end for

display topics with more than MIN_GROUPS_PER_TOPIC authors
```

First, documents are grouped by authors. If a document has multiple authors, it is considered to be associated with all of them and the document is duplicated in the result set. Since the LDA computed the topic affiliations of each document, we can now calculate the topic affiliations of each author by simply averaging the affiliations of each document belonging to that author. An author is then only considered to be affiliated with a topic if the average affiliation is more than one third (`MIN_AFFILIATION = 0.33`). One third was chosen as the result of a comprehensive testing phase, where 33% proved to give the right balance between topics growing too large and important authors of a topic not showing up.

To make sure an author is only affiliated with topics he is important for, the `MAX_TOPICS` constant determines the maximum number of topics an author can belong to. Since we set `MIN_AFFILIATION = 0.33`, this step proved unnecessary. With this `MIN_AFFILIATION`, an author can only be affiliated with a maximum of three topics. So in the final prototype, we set `MAX_TOPICS =` ∞.

Finally, we reduce the number of topics displayed on the screen by pruning topics that contain only a small number of groups. If a query returns very few documents for a certain topic, we can consider that topic not worth showing up in the result visualisation. For simplicity reasons, in our prototype we used a fixed value of `MIN_GROUPS_PER_TOPIC = 5`.

The final result of this algorithm are topics with authors assigned to them that contributed considerably to that topic. Additionally, we make sure that topics that contain very few results are not displayed. The topics are displayed on the screen with their top three words when zoomed out (see Figure 16 on page 41). Additionally, the number of authors and the number of documents contained in a topic are displayed in the upper right of the topic. If enough space is available, the top fifty words of a topic are displayed in a word cloud, where the font size of each word reflects its importance (see Figure 17 on page 42). When the user decides to explore a topic, relevant authors for that topic are displayed as we will further describe in Section 4.5.2 on the next page.

Grouping topics The topics are placed on a ZOIL-based, two-dimensional landscape and can be moved, resized and zoomed into freely with multi-touch gestures. The user can zoom into one or multiple topics to explore them further. This allows the user to make use of spatial relationships to group topics, for example by discarding all irrelevant topics by putting them on a pile in one corner of the landscape. Additionally, we developed an algorithm that makes use of the spatial possibilities by grouping similar topics together. We do this by comparing the terms that define a topic: If those terms have a large overlap, topics are considered to revolve around a similar theme and are therefore grouped together spatially. As it can be seen in Figure 17 on page 42, there is one cluster in the top left of the landscape that contains four topics, two clusters with two topics each and three single topics that are not contained in any cluster. Making use of real-world knowledge of users they can probably conduct that each cluster seems to revolve around a certain, more general theme: The large cluster contains topics about computer science, while the two small clusters are about biology and political science. With large result sets, this grouping allows the user to get an even more abstract view on the data. After picking a cluster of interest, the user can then further refine the search results by picking one or multiple topics to look at. This all happens implicitly by browsing the data space.

With this representation, results are broken down down into something the user can understand and that is of interest to the user - he can dig into results about the topic he is researching and ignore the rest of the result set, implicitly filtering the data space.

4.5.2 Exploring a topic

When the user decides to explore one or more topics, pinching gestures or tapping a topic will zoom into it. Thanks to semantic zooming, the user stays completely in his current context of work - he can move around on the same landscape with the same topics - while drilling down. After showing only their three most important words at first, as it can be seen in Figure 16 on page 41, the topics will then reveal a word cloud with their fifty most important words (see Figure 17 on page 42). When the user zooms in even further, the topics then reveal their content. As previously discussed, a topic can contain a large number of documents, which is why we introduced another abstraction before individual documents: As it can be seen in Figure 18, the documents of a cluster are assigned to their respective authors and these authors are then displayed.

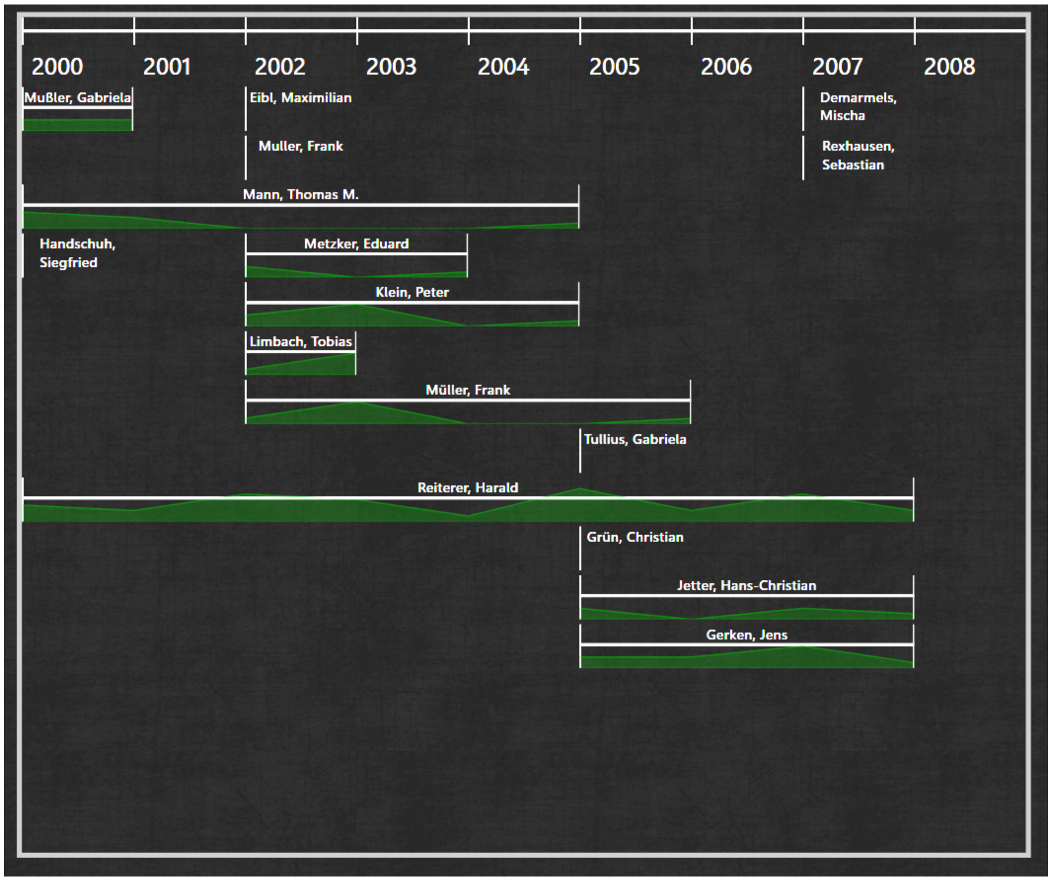


Figure 18: The topic “information, user, interaction, ...” seen in Figure 17 on page 42, zoomed in to reveal more detail. It shows authors that wrote in that topic. The width of an author determines when the author was active in the topic, the line chart shows the amount of published work per year. Additionally, authors further to the top are considered more affiliated with the topic.

The landscape was used to give the position of each author a meaning in both x- and y-direction. A timeline spans the x-axis and authors are put on that timeline according to when they worked in the explored topic. Additionally, a line chart is drawn for each author that allows the user to see how many publications the author made each year. This way, a user can easily see when an author stopped working for a while, moved to another field, which authors are constantly publishing in the topic and which authors rarely publish. This allows a user to quickly determine important, active authors, even if the user is not familiar with the topic and important people working in that field. For example, in Figure 18 on the previous page we see the topic "information, user, interaction, ..." (from Figure 17 on page 42) zoomed into. The placement of authors on the x-axis shows that "Frank Müller" first published in the topic in 2002 and his latest publication in the topic is from 2006. It also shows that he was not very active in recent years, but was so at the beginning of his work.

To make topics more comparable they share the same timeline, so the first and last year of the timeline is identical for all topics. The timeline allows users to explore important authors first and move to less active authors later or when finding interesting publications where they were involved. Since the affiliation of each author with the topic was computed, we used this as a second measure for importance. The authors are drawn on the y-axis according to the LDA-based affiliation score with more affiliated authors further to the top. The ordering in y-direction is not stable, because the algorithm is space efficient without producing overlap between authors before taking topic affiliation into account. Still, authors nearer to the top of the topic are much more likely to be more affiliated with the topic than authors further down the axis. We will now go into detail on how the authors are positioned by first introducing the pseudo-code that performs the positioning and then describing it in more detail.

Listing 3 Algorithm that positions authors inside a topic. It prevents overlapping and considers topic affiliation for each author.

```
Input: globalEarliest and globalLatest - earliest and latest publication year in all topics
Input: groups - the author-groups as produced by Listing 2 on page 43
Output: Authors, displayed at their position inside a topic

sortedGroups ← groups sorted by affiliation to current topic
highestY ← new array
for each group in sortedGroups do
    earliest ← earliest year the group author wrote something
    latest ← latest year the group author wrote something
    y ← 0
    // Author will be drawn from earliest to latest on the timeline
    // highestY keeps track where authors were drawn already - we use this to
prevent overlap
    for each year from earliest to latest do
        if highestY[year] exists and highestY[year] > y then
            y ← highestY[year]
        end if
    end for
    x ← (earliest - globalEarliest) * YEAR_WIDTH
    draw group at x, y
    y ← y + GROUP_HEIGHT
    for each year from earliest to latest do
        highestY[year] ← y
    end for
end for
```

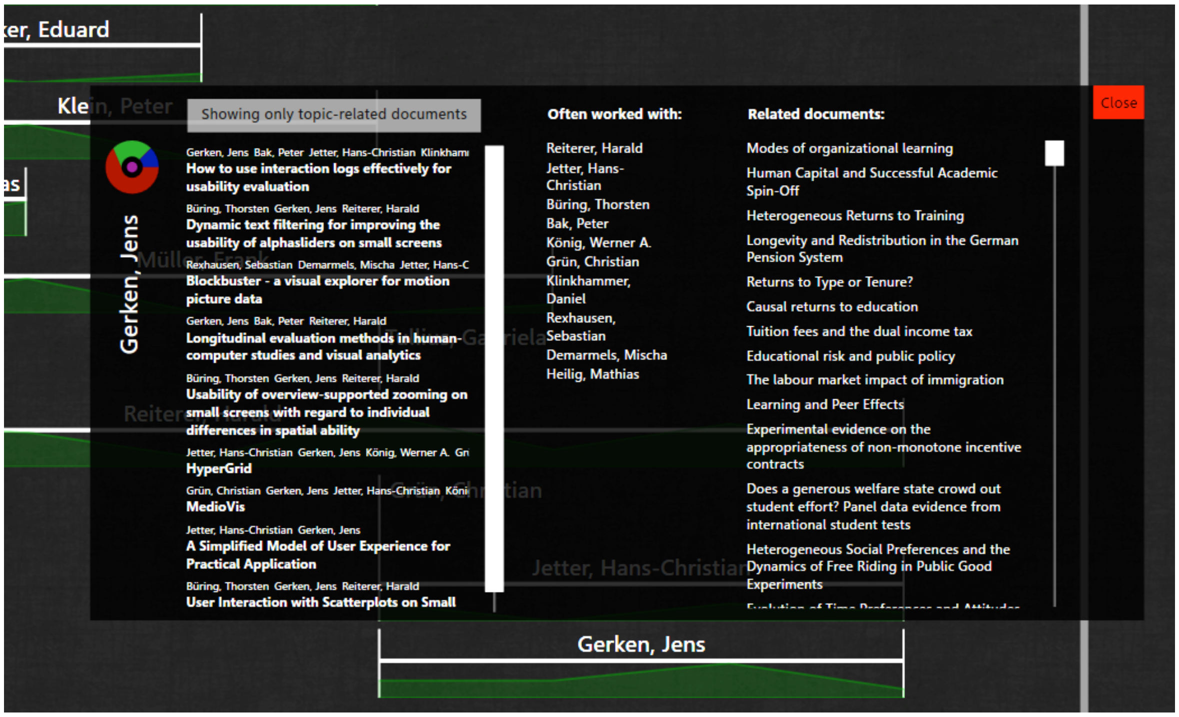
Authors are ordered by their affiliation with the topic. This means that authors are handled in the order of their affiliation, which increases the likelihood that more affiliated authors appear further to the top, because the landscape is still empty and the chance of being pushed to the bottom because of overlaps is smaller. To prevent such overlap the algorithm remembers the highest y position already drawn at for each year. For each author, each year he spans is checked for overlap - if the author would overlap another author in any of the years, the y position is increased until the author can be drawn without any overlap. The algorithm performs quite well in practice and is quick and suitable for the task at hand. The final result will be a timeline were authors are placed on the time axis and authors at the top are more likely to be relevant to the topic than authors at the bottom.

With this arrangement, we give the user the ability to get a feeling for the topic - which persons are important for it, work a lot in it, publish in it a lot and more with only a few glances. We help the user in determining important authors and therefore make it easier for the user to decide where to drill down further into the data-space and explore relevant authors.

4.5.3 Exploring an author

When the user decides for one more interesting authors, he can then tap them to open up the author detail view. This differs from the interaction used so far, as tapping an author does not result in a semantic zoom. It instead opens a popup, as it can be seen in Figure 19 on the following page. While the popup is still staying in the context of the application and is embedded into the landscape - and can therefore be zoomed and moved like all other objects - there are multiple reasons why we did not rely on semantic zooming in this case:

- Interaction in a very deep zoom level proved difficult: Some might expect a single-finger pan to move the object they are currently zoomed into, others might expect the same gesture to result in panning of the landscape.
- There are high technical challenges in providing support for deep semantic zooming where the solutions exceeded the scope of a prototype.
- We observed that zooming too far into the landscape made users lose their sense of context, a phenomenon described by Reiterer et al. as "lost in hyperspace"[65, p. 4]. At some point, semantic zooming therefore lost its main advantage: Staying in context.
- We wanted the user to be able to quickly compare authors with each other, for example to see if they had documents in common. Since authors have a fixed position inside their topic, as determined by the algorithm described in Section 4.5.2 on page 45, they can not be dragged next to each other for comparison. Popups can be moved next to each other for comparison of multiple authors, even across topics.



Gerken, Jens Bak, Peter Jetter, Hans-Christian Klinkhamı
How to use interaction logs effectively for usability evaluation

Büring, Thorsten Gerken, Jens Reiterer, Harald
Dynamic text filtering for improving the usability of alphasliders on small screens

Rexhausen, Sebastian Demarmels, Mischa Jetter, Hans-C
Blockbuster - a visual explorer for motion picture data

Gerken, Jens Bak, Peter Reiterer, Harald
Longitudinal evaluation methods in human-computer studies and visual analytics

Büring, Thorsten Gerken, Jens Reiterer, Harald
Usability of overview-supported zooming on small screens with regard to individual differences in spatial ability

Jetter, Hans-Christian Gerken, Jens König, Werner A. Gri
HyperGrid

Grün, Christian Gerken, Jens Jetter, Hans-Christian Köni
MedioVis

Jetter, Hans-Christian Gerken, Jens
A Simplified Model of User Experience for Practical Application

Büring, Thorsten Gerken, Jens Reiterer, Harald
User Interaction with Scatterplots on Small

Often worked with:

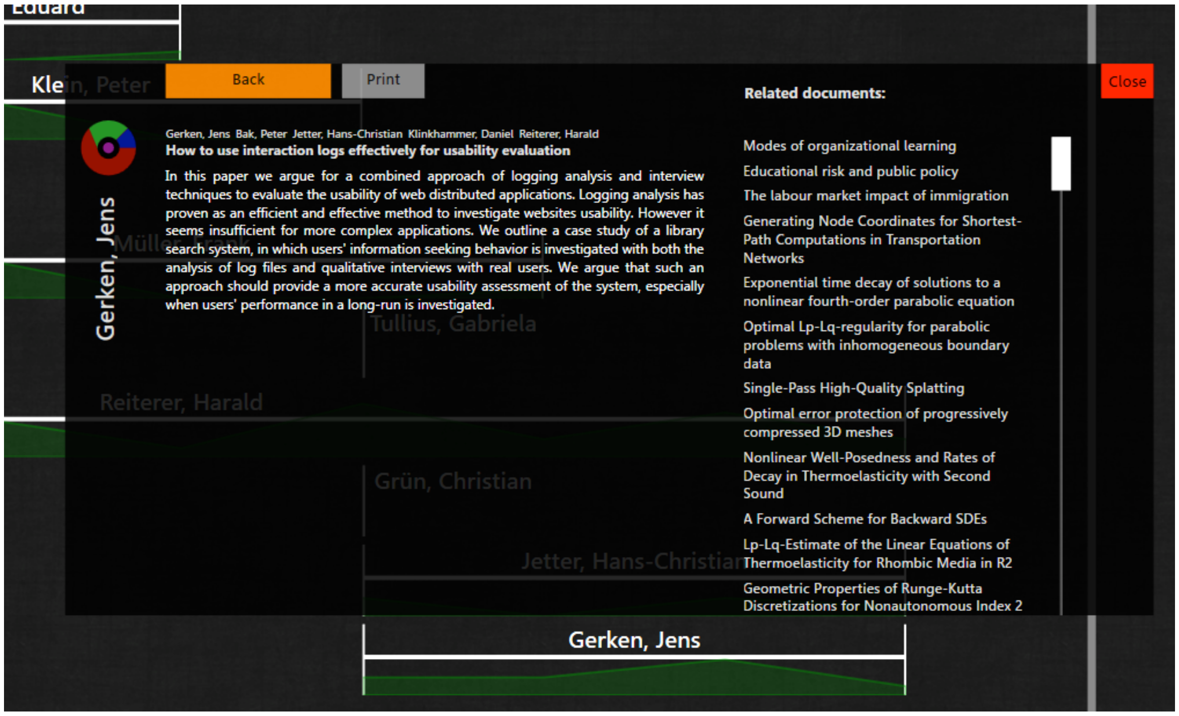
Reiterer, Harald
Jetter, Hans-Christian
Büring, Thorsten
Bak, Peter
König, Werner A.
Grün, Christian
Klinkhammer, Daniel
Rexhausen, Sebastian
Demarmels, Mischa
Heilig, Mathias

Related documents:

Modes of organizational learning
Human Capital and Successful Academic Spin-Off
Heterogeneous Returns to Training
Longevity and Redistribution in the German Pension System
Returns to Type or Tenure?
Causal returns to education
Tuition fees and the dual income tax
Educational risk and public policy
The labour market impact of immigration
Learning and Peer Effects
Experimental evidence on the appropriateness of non-monotone incentive contracts
Does a generous welfare state crowd out student effort? Panel data evidence from international student tests
Heterogeneous Social Preferences and the Dynamics of Free Riding in Public Good Experiments

Figure 19: The details view of a single author. It pops up when an author is tapped in the timeline (Figure 18). It shows the author's documents in that topic, related authors and related documents. It furthermore features the ChromeViz in the top left, which we will take a deeper look at in Section 4.5.4 on the next page. Tapping a document allows the user to display the document's abstract and related documents.

Regardless of the alternative interaction, the content of the author detail view is what would be expected: It allows the user to explore an author in detail. The view shows the documents of the author related to the explored topic, common co-authors, related documents and allows to switch to an all-document list where all documents of the author are shown, including those not related to the current topic. Individual documents can be tapped, which leads to a view like it can be seen in Figure 20 on the following page, where the abstract of a document can be viewed along with the documents authors and related documents. Furthermore, a circular glyph can be seen in the top left corner, above the author name. This is a custom-developed glyph called ChromeViz. Building the application with *FacetSearch++* support (as explained in Section 4.4.1 on page 35) allowed us to have up to three different search queries sent to our system at the same time. ChromeViz gives information about how the author's publications relate to each of the result sets and how many of the author's documents are in multiple result sets at once. We will take a more detailed look at this glyph in the following section.



Gerken, Jens Bak, Peter Jetter, Hans-Christian Klinkhammer, Daniel Reiterer, Harald
How to use interaction logs effectively for usability evaluation
In this paper we argue for a combined approach of logging analysis and interview techniques to evaluate the usability of web distributed applications. Logging analysis has proven as an efficient and effective method to investigate websites usability. However it seems insufficient for more complex applications. We outline a case study of a library search system, in which users' information seeking behavior is investigated with both the analysis of log files and qualitative interviews with real users. We argue that such an approach should provide a more accurate usability assessment of the system, especially when users' performance in a long-run is investigated.

Related documents:
Modes of organizational learning
Educational risk and public policy
The labour market impact of immigration
Generating Node Coordinates for Shortest-Path Computations in Transportation Networks
Exponential time decay of solutions to a nonlinear fourth-order parabolic equation
Optimal Lp-Lq-regularity for parabolic problems with inhomogeneous boundary data
Single-Pass High-Quality Splatting
Optimal error protection of progressively compressed 3D meshes
Nonlinear Well-Posedness and Rates of Decay in Thermoelasticity with Second Sound
A Forward Scheme for Backward SDEs
Lp-Lq-Estimate of the Linear Equations of Thermoelasticity for Rhombic Media in R2
Geometric Properties of Runge-Kutta Discretizations for Nonautonomous Index 2

Figure 20: The author detail view after an individual document was tapped, displaying information about that document like abstract, authors and related documents.

4.5.4 Displaying multiple result sets

Additionally to the possibilities in exploring the data space, supporting *FacetSearch++* gave us an interesting opportunity. Search result sets are sent to our prototype by putting tokens inside the streams of *FacetSearch++*. We extended this idea by adding support for multiple tokens/result sets simultaneously to our application. When the user puts more than one token inside the streams of *FacetSearch++*, our prototype picks up all the result sets and merges them together. This allows the user to pursue different search strategies and explore all of them simultaneously. For example, the user could search for different authors, different keywords or even two entirely different areas of expertise. Additionally, the user should have the opportunity to discover how the different result sets influence his merged result. For this reason, we developed ChromeViz, a glyph that allows the user to quickly see and compare how different result sets are related to each other. ChromeViz is named after the Google Chrome browser logo[35], which sparked the initial idea for this visualisation. It is displayed in the author detail view, as seen in Figure 19 on the preceding page, and displays how the different documents of the author are distributed over the different result sets. In the following, we will show the construction and design decisions leading to ChromeViz.

ChromeViz solves the problem of displaying differences and similarities of up to three result sets. This includes finding out how large each result set is and if documents occur in more than one result set. Because of this, the requirements to the glyph were quite strict, the following requirements should be fulfilled:

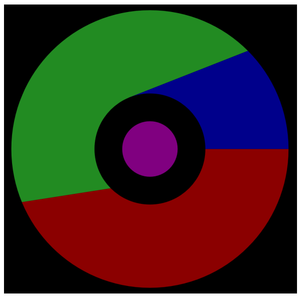


Figure 21: An example ChromeViz glyph. Each segment represents one result set and the purple circle in the middle represents the intersection of all three result sets.

- It should be easily understandable.
- It should be comparable. It is more important that multiple instances of this visualisation can be compared to each other easily than to get exact numbers out of it.
- It should be space-efficient. Since the ChromeViz will be embedded into other views, the size it works on should be adjustable but the glyph should still work at small sizes. The minimum targeted size was 50 by 50 pixels.
- The glyph should work for one, two or three result sets.
- The intersection of two result sets must be visible for any possible combination.
- The intersection of all three result sets must be visible.

Fulfilling these requirements proved more difficult than originally thought and standard visualisations were not suitable for the task, because trying to encode the required information into a standard visualisation, for example a bar chart, required too much space and was not comparable enough, especially when far away from each other on the screen. For this reason a custom visualisation was developed to solve the problem. It became clear that a circle-like shape seemed the most fitting for this task: When creating an ordinary pie chart with three segments - one segment for each result set - there naturally exists a border between any two segments that could be used to visualise overlap between the segments. So the ChromeViz is, on a basic level, a modified pie chart. A pie chart allows the user to easily see the size of each segment (result set). The ChromeViz extends this to visualise the overlap between each combination of result sets and also the overlap between all three result sets. It should be noted that the glyph works with a maximum of three result sets - more tokens are not accepted by our prototype if put onto the *FacetSearch++* streams. After putting multiple coloured tokens on *FacetSearch++*, looking at a ChromeViz instance allows the user to quickly assess how documents are distributed over the different result sets. This is done by comparing the sizes of the individual segments of the circle. Each segment represents the result set of the token of the same colour. If the visualisation displays only a single colour, an author wrote only documents in that result set. If more segments are displayed, the user immediately knows the author has documents in all those result sets. The sizes of the segments show the user the amount of documents in each result set. Additionally, as seen in Figure 22 and Figure 23 on the next page, the overlap between two result sets is visualised by one segment “bleeding over” into another. In the shown figures, this is especially notable between the green and blue segment - in Figure 22 both segments have a large number of documents in common, therefore the overlap between both segments is large. In Figure 23 the result sets have no documents in common and therefore the border between both segments is “hard”.

The amount of documents that occur in all three result sets is visualised by the size of the purple circle at the center. If the circle is not visible, there are no documents that occur in all three result sets (or there aren’t three tokens on *FacetSearch++*). As it can be seen in Figure 24 on the following page, the purple circle increases in size when there are many documents that are in all three result sets. If all documents would be in all three result sets, the entire visualisation would be purple and no individual segments would be visible at all.

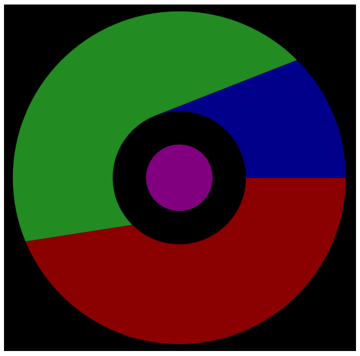


Figure 22: An example ChromeViz glyph where the result sets represented by the green and blue segment have a lot of documents in common, while blue and red have almost no, and green and red only a few documents in common.

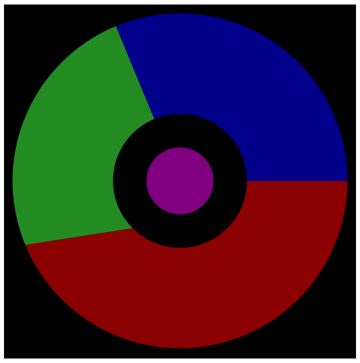


Figure 23: An example ChromeViz glyph. In contrast to Figure 22, the green and blue segment share almost no documents.

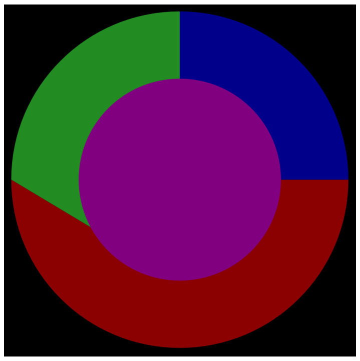


Figure 24: An example ChromeViz glyph where there is a large amount of overlap between all three result sets, represented by a large purple circle in the center.

In Figure 25 we see the geometric shapes that are used internally to construct ChromeViz. In yellow we see the *normal circle*. It has a radius of `r` and is where the final visualisation will end up. This also means that the visualisation will have a bounding rectangle with `2r * 2r` in size, which is represented by the white rectangle. Additionally, we calculate the circumcircle around that rectangle, called *outer circle*. We will go into detail on why we need this circle later. Additionally, we define the *inner circle*, which is the black hole at the center, as depicted in Figure 21 on the preceding page. The size of this is arbitrarily chosen - the bigger this circle, the better the "bleeding effect" will be visible but the less of the coloured segments will be seen. If the inner circle would be 0 in size, this "bleeding effect" would not be achievable at all and overlap between result sets could not be visualised.

With those geometric shapes defined, we developed an algorithm for drawing ChromeViz. We will post a simplified version of the algorithm here that will then be explained in detail in the following.

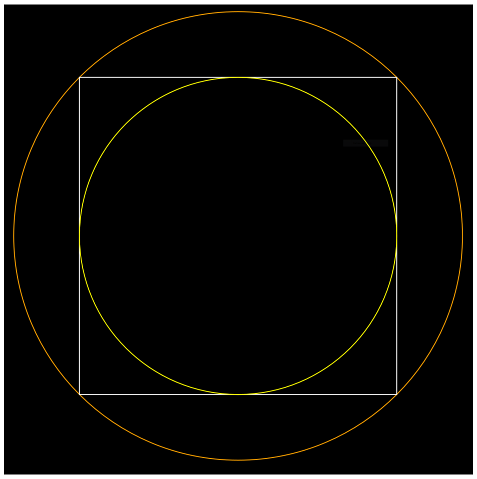


Figure 25: The geometric shapes needed to construct ChromeViz. Our final visualisation will be drawn inside the normal circle (yellow). Additionally, we have a surrounding rectangle (white) and a circumcircle around that rectangle (orange).

Listing 4 Algorithm that draws ChromeViz. It draws a segment for each of the possible three result sets, then draws the purple all-token circle and finally clips to the normal circle.

```
Input: One to three search result sets
Output: ChromeViz glyph displayed on screen

currentPercentage ← 0
for each result set do
    thisSetPercentage ← fraction of documents unique in the current result set
    thisAndLastSetPercentage ← fraction of documents intersecting current and last result set
    thisAndNextSetPercentage ← fraction of documents intersecting current and next result set

    // Calculate points where last segment ended = start of current segment
    lastInner ← point on inner circle at 360° * (currentPercentage - thisAndLastSetPercentage)
    lastNormal ← point on normal circle at 360° * currentPercentage
    lastOuter ← point on outer circle at 360° * currentPercentage

    // Calculate points where the current segment ends
    currentOuter ← point on outer circle at 360° * (currentPercentage + thisSetPercentage +
thisAndNextSetPercentage)
    currentNormal ← point on normal circle at 360° * (currentPercentage + thisSetPercentage +
thisAndNextSetPercentage)
    currentInner ← point on inner circle at 360° * (currentPercentage + thisSetPercentage)

    // Prevent artefacts by creating a vertex on the outer circle every 90 degrees
    checkpoints ← empty
    checkpointPercentage ← currentPercentage + 0.25                    // 0.25% = 90°
    while checkpointPercentage <= currentPercentage + thisSetPercentage do
        add point on outer circle at 360° * checkpointPercentage to checkpoints
        checkpointPercentage ← checkpointPercentage + 0.25
    end while

    // Draw polygon
    draw polygon from lastInner to lastNormal to lastOuter to all checkpoints to currentOuter to
currentNormal to currentInner
    fill polygon with token colour
    currentPercentage ← currentPercentage + thisSetPercentage + thisAndNextSetPercentage
end for

// Draw purple all-tokens circle
allSetsPercentage ← fraction of documents intersecting all three result sets
draw circle at the center with radius r*allSetsPercentage
fill that circle with purple

clip everything to shape of normal circle
```

The algorithm first walks over every result set - a minimum of one and a maximum of three sets are allowed. For every set, a polygon is calculated and drawn that is later clipped to the shape of the normal circle. Just drawing a segment itself, without "bleeding effect", would be very simple. To achieve that, the segment's size is adjusted according to the size of the result set it represents. The first vertex of the polygon would be drawn at the last vertex of the last segment, or at 0^o for the first segment. We would then "walk" as many degrees on the circle as the segment has documents. For example, if the result set of the red token contains 30% of the entire results, we would walk 108^o on the boundaries of the normal circle from our starting position and put another vertex there. Additionally, vertexes would be put on the boundaries of the inner and outer circle at the same position. To make sure that artefacts as in Figure 26 are prevented, checkpoints are created on the outer circle, each checkpoint being an additional vertex. The checkpoints will be added if the current segment is larger than 90^o and will be added at least every 90^o on the boundary of the outer circle.

With the segments set up, the "bleeding effect" has to be added to that. To do so, we use a similar strategy: On the inner circle, we do exactly what we just described - calculate the amount of degrees the current segment should have based on the number of documents it contains and "walk" that amount of degrees on the inner circle, starting at the segments start position. On the outer circle, we do the same, but additionally "walk" as many degrees as the fraction of the documents the current result set has in common with the next result set in the circle.

An Example: We start at a position of 0^o and with the red segment. We start creating the polygon for the red segment and put the first three vertices at the starting position - at 0^o on the inner, normal and outer circles. In C#, 0^o is defined as the right side of the circle. Assuming that the result set of the red token makes up 30% of the entire results, we will walk $360^o * 0.3 = 108^o$ on the circles, starting at 0^o. Simply putting another three vertices here would produce an output similar to Figure 26. To prevent this, we add a checkpoint on the outer circle every 90^o from our starting position - in this case this means at the very bottom, at 90^o. This then produces a result like in Figure 27 on the next page. We now determine that the next token will be the green token and that the result sets of the red and green token have 15% of the entire results in common. This means that we walk an additional $360^o * 0.15 = 54^o$ on the normal and outer circle, which then produces a result as in Figure 28 on the following page. We follow this procedure with the green (Figure 29 and Figure 30) and blue (Figure 31) segments. When finished, we clip the visualisation to the normal circle (Figure 32) and add the purple all-tokens circle if necessary (which has the same center as the normal circle and is sized $r * \{\text{fraction of documents in all three result sets}\}$). This will produce the final glyph.

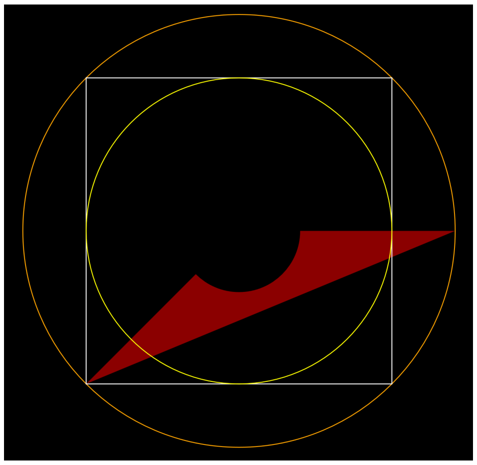


Figure 26: A wrongly drawn segment because the outer circle "checkpoints" were missing.

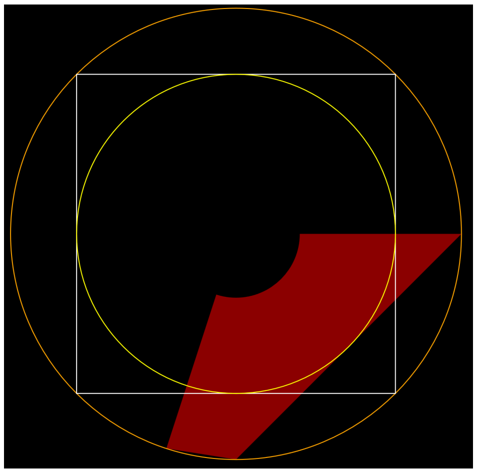


Figure 27: ChromeViz during construction. Only the red segment was drawn so far.

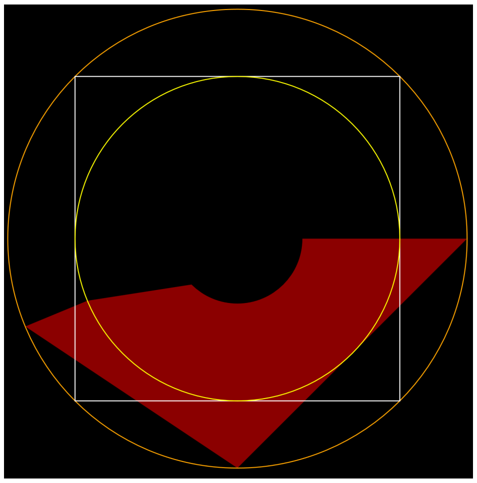


Figure 28: ChromeViz during construction. The red segment including its overlap with the next segment (not yet drawn) was drawn so far.

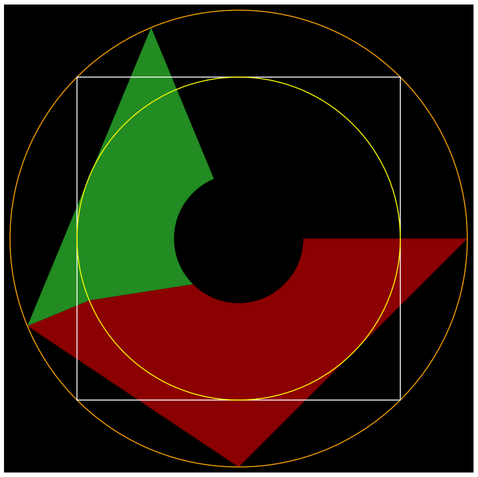


Figure 29: ChromeViz during construction. The red and green segments were drawn so far.

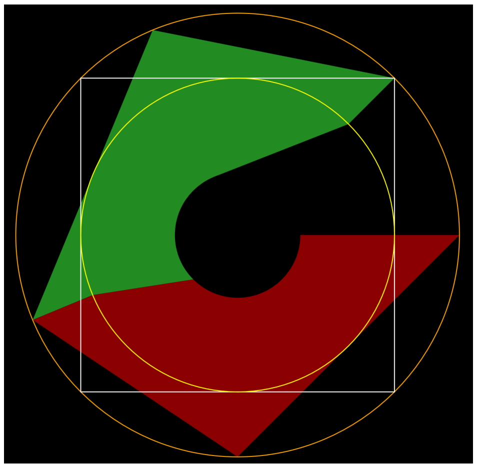


Figure 30: ChromeViz during construction. The red and green segments, including their overlap with the next segment (not yet drawn), were drawn so far.

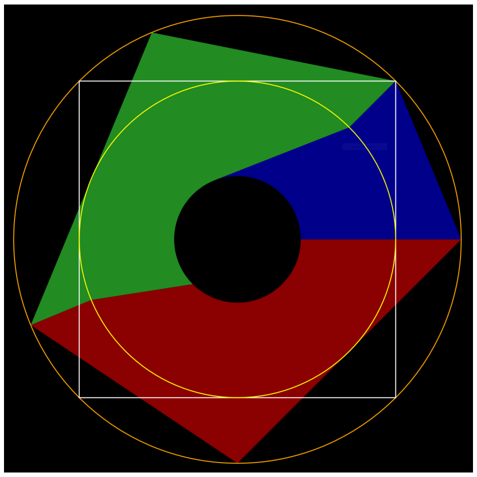


Figure 31: ChromeViz during construction. All segments have been drawn, but not yet clipped.

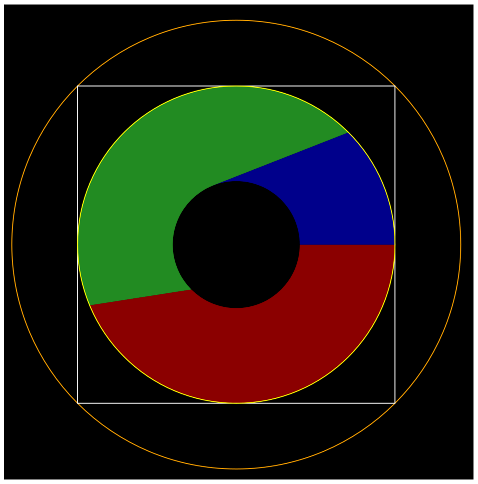


Figure 32: The same visualisation as in Figure 31, but clipped to the normal circle. The purple all-token circle has not been drawn yet and the geometry lines have to be removed to produce the final result.

In conclusion, ChromeViz visualises all the desired information, can be scaled to almost any size and is comparable over larger distances as they might occur in a landscape-based system.

4.6 Sense-making

As described in the previous sections, our prototype allows a user to drill down into a search result, filter the search results implicitly by selecting one or more relevant topics for further exploration and then allows the user to find important authors, co-authors and documents. We got from a complex, large search result down to individual documents, all by exploring the data space. When the relevant parts of the result space have been identified by the user, the need arises to make sense of those parts. A lot of authors pointed out the importance of the sense-making step. For example, Evans and Chi said it was an important part of the *informational searches*[20] and Pirolli and Card dedicated a huge amount of attention to this part of the search[59]. Sense-making can, for example, include skimming or reading of interesting documents in more depth and can include annotating them, sorting them by different criteria, summarising them etc.

To support sense-making, we added the *Personal Landscape*, a second landscape that co-exists with the search result landscape. The user is able to add relevant parts of the search result to the Personal Landscape, making those parts permanent even when a new search is conducted. For example, authors and entire topics can be added to the Personal Landscape. To add something to the Personal Landscape, the user can touch it and drag it across the screen, which will make the *Personal Drop Zone* appear at the bottom of the screen, as seen in Figure 33.

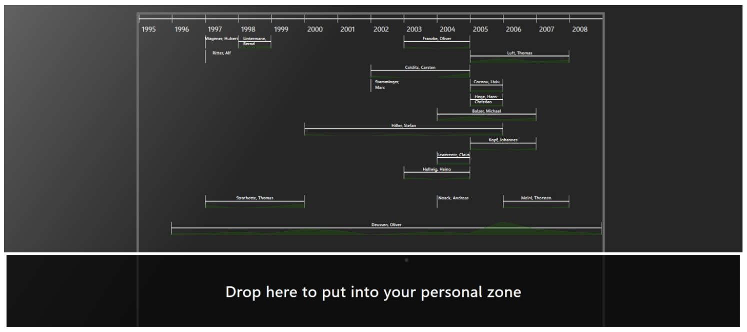


Figure 33: When the user starts dragging an object that can be added to the Personal Landscape, the "Personal Drop Zone" appears at the bottom of the screen. Dragging the object onto it and releasing it will add it to the Personal Landscape.

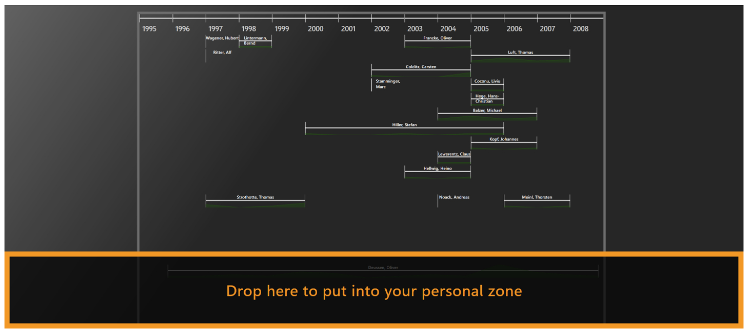


Figure 34: The "Personal Drop Zone" in its active state, when a droppable object is above it. Dropping the object will add it to the Personal Landscape.

Dragging the object above the "Personal Drop Zone" will transform the Zone to an active visual state (see Figure 34), and a drop will then result in the object being added to the Personal Landscape. Additionally, the object is marked as personalised by adding a border in the same colour as the active "Personal Drop Zone" and is returned to the position it had before the drag started. Objects can be added to the Personal Landscape multiple times, but once marked as personalised they will stay that way. We are confident that the integrated fashion of the Personal Landscape and the sense-making process itself will be of great use to knowledge workers. Furthermore, the switch between goal-oriented analysis and discovery-based browsing explained by Gerken et al.[25] is supported by this: The user can switch between the search results (browsing) and making sense of the relevant results (analysis) easily, quickly and without switching to another application. To see the Personal Landscape at any time, the user can simply zoom out. The Personal Landscape will be found next to the search result landscape. The user is free to explore the Personal Landscape by panning and zooming exactly the same way as he is with the search results. Additionally, though, we added a number of sense-making possibilities to aid the user in his sense-making process:

4.6.1 Spatial arrangement

Similar to the search results landscape, objects on the Personal Landscape are freely movable. This allows the user to use spatial relationship to bring his own order to the objects. For example, authors that worked a lot together can be grouped in one corner of the landscape or results from the same search can be grouped together and so on. Objects could also be grouped by topic, having a topic and all objects that belong to that topic in one corner, and another topic with its objects in another and so on.

Furthermore, the Personal Landscape automatically expands when the users drags an object to its edge. This allows users to make the Personal Landscape as big as they desire and does not restrict them in their sense-making process.

4.6.2 Sizing

Similar to the spatial arrangement, the objects on the Personal Landscape are resizable. This allows the user to use size to encode information. For example, the user is able to make more important authors or topics bigger, so their increased importance can easily be recognised, even when zoomed out.

4.6.3 Exploration

One important technical fact that we wanted to assure is that the objects in the Personal Landscape stay fully explorable. For example, an added topic will still show the word-clouds and timeline with authors when zoomed into and the authors can still be tapped to get to the author details views. This allows the user to look up why he added something, what it contains and allows him to retrieve all the information of an object from the Personal Landscape, even after the search result landscape has been cleared and filled with new results. It also allows the user to backtrack his work, figuring out what he did and why he added something to the Personal Landscape.

4.6.4 Linking

One of the most basic but most important sense-making abilities is the expression of relationships, which is for example the basic idea behind mind-maps. The user may wish to express relationships between objects on the Personal Landscape. As mentioned, one possibility to do this is spatial arrangement. But arranging a high number of objects can be tedious and may not fit the user's mental concept of the Personal Landscape. To solve this, we implemented a feature specifically made for expressing relationships. The user can draw a line between two objects to create a link between them, which will visually link the objects and move with the objects when they are moved or resized.

Since annotating is a special task not performed anywhere else in the system, its interaction should be clearly distinguished from other interaction techniques. The linking gesture is therefore performed with a touch-pen provided with the application. The user can use the touch-pen like he would use a finger, simply dragging from one object to another with the pen instead of his hand. The application will pick up the pen movement and distinguish it from a hand gesture. Furthermore, this keeps both, annotation and non-annotation gestures, simple and easy to remember: Instead of creating an entirely new gesture for linking, a simple drag-gesture is used, but with a different device. The application can then easily distinguish a panning gesture from a linking gesture. Additionally, the pen is a real-world resemblance to annotation and note-taking. This allows the user to easily remember when to use the pen (for annotating) and when to user his finger (for moving in the data space).

4.6.5 Drawing

Like expressing relationships, taking notes to express the thoughts and feelings during the search process is very important in sense-making. It allows the user to remember his thoughts on certain aspects of the result set which helps him to remember findings and knowledge that he already developed during the search process, as well as encode information to develop new knowledge. For this prototype, we decided to focus on annotations outside of documents. In-document annotations are an important and very large field of research and were therefore out of scope to be additionally explored in full detail in this project and thesis.

On the Personal Landscape, the user is able to structure objects and take down his thoughts on objects. Like linking, this is done with a touch-pen in order to represent the note-taking character of this interaction and clearly distinguish it from other interaction techniques. Actually taking down notes is as simple as possible: The user can draw or write everywhere on the Personal Landscape. This means that, using a touch-pen, the user is able to take notes, create mind-maps, write down important authors, documents or other findings and so forth just like he would on an ordinary piece of paper. This will help the user in easily getting into the annotational process without having to switch to another application or having to deal with complex technical issues like selecting a pen, thickness, stroke etc. The real-world knowledge from pen & paper interaction can be directly applied to our system in order to annotate the Personal Landscape.

4.7 Conclusion

Our concepts, and with it our prototype VIS system, will make a useful addition to the existing concepts of the Blended Library. Our goal was to support the process of expressing the information need, exploring the result sets and making sense of the parts that were found to be relevant. Our prototype allows the user to get an abstract view on the search results by breaking the documents down to authors, topics and then clustering those topics by thematic relationship. The user can explore all these abstractions within an integrated context and workflow. A movable, two-dimensional landscape and the concept of semantic zooming allow the user to freely switch between the different abstractions and views with simple multi-touch gestures. The system furthermore allows the user to explore individual authors and documents and save interesting parts of the search results in the Personal Landscape, which saves objects across multiple searches. The Personal Landscape gives an integrated support for sense-making, allowing the user to encode information using spatial arrangement, sizing, linking or free annotations and notes with a pen. Our goal was to support searching & sense-making in a way that is closer to the way a usual library worker searches for information. We will now put parts of our concepts to the test by comparing our prototype to an ordinary library search system in a comparative study.

CHAPTER 5

EVALUATION

In this chapter, we will take a look at the user study conducted after implementing our prototype system to evaluate our concepts. Because evaluating all of the concepts in a single study would lead to a huge number of influencing factors, only selected concepts where evaluated in this study. Additional study would have to be conducted to test other parts of the system.

In this study, we wanted to see how users use the prototype to explore a previously unknown topic and how they made sense of that topic using our integrated sense-making tools. As control condition, users had to solve the same tasks with KOPS[57], the usual search tool for online publications in the library of the University of Konstanz. KOPS is a web interface, allowing users to search for publications by keyword, returning textual results to them (see Figure 35 on page 61). Users can then click on individual publications to see author, year, abstract and sometimes attached keywords if present.

Design & Participants We used a double counter-balanced within-subjects design with 8 participants. The within-subjects design was chosen because the task of researching a topic and making sense of it can be very subjective - comparing the results of different participants is difficult and only a subjective comparison is possible. Using the within-subjects design allowed us to compare the test system and the control condition of the same subject. To ensure the study was not biased, each participant was given two different topics to research, so the knowledge gathered in the first run did not influence the second. Furthermore, we double counter-balanced the study with the two uncontrollable influences we identified: The order in which the systems were tested and the topic the users had to research. Counter-balancing the order of the systems mad sure participants were not preferring one system over the other because of the order in which they were tested and additionally countered effects like fatigue after the first run. Counter-balancing the topics made sure that differences in the difficulty of the topics do not influence the result and no combination of a certain topic with a certain system was easier or harder then other combinations.

Our participants were students or people in job training; students of computer science were not allowed in the study. The average age was 23 years (SD = 2.75 years). Additionally, participants were asked about their computer usage and most were computer savvy. When asked “How many years have you been using a computer?” the average answer was 11 (SD = 4.21 years) and when asked to judge their own computer savviness on a scale from 1 (“Beginner”) to 5 (“Expert”), the average answer was 3.1 (SD = 0.8). Furthermore, all of them said they had used touch screen devices before.

Tasks & Procedure Our task required participants to research a topic that was previously completely unknown to them. Participants were given 30 minutes per system for their research. The goal of the task was to get a first grasp of the topic, and conduct in a first, broad sense-making step as it is often conducted by researchers at the beginning of a new task. The goal was to find aspects, for example authors and key terms, that would be worth further investigation in the future. To guide users through this, they were given four steps they could use as a help to solve this task:

1. Find authors that, in your opinion, are important or in some way influential to the topic.
2. Find terms or subjects the chosen authors seem to write about often.
3. Are there any other terms you think you would need more information about in order to understand the topic?
4. Find publications that, in your opinion, are important or in some way influential to the topic or the authors you chose.

The task was deliberately chosen openly, as we wanted to simulate an open research as it is often conducted in *informational searches* and see how participants would approach this task. Participants had to solve this task with two different search queries, once with KOPS and once with our prototype system. The two search queries used were "human computer interaction" and "brain picture processing". As mentioned before, the query was one of the two factors that were counter-balanced, so the combination of system and query was changed for each participant. Furthermore, in the prototype, participants could use the entire search result landscape, but they were given one of the displayed topics as the target for their research.

Prior to working on the task, users were given an introduction to our system. The features and interaction was shown to them and they were also introduced to the Personal Landscape and how to use the pen as input. They then were given a shot amount of time of training, usually less than five minutes, until they felt ready to start. Users then started the task with the first query on the first system with a thirty minute time limit and performed the same tasks with the other query on the other system afterwards with the same time limit. After each system, users were asked to fill out two short questionnaires about their subjective experiences with the system. When working with KOPS, users were asked to put down their findings on pen & paper. In the prototype, users were asked to use the Personal Landscape to take down their findings. Participants were compensated with 8 EUR for their time.

Results From observing participants during the study and the results they created in both systems in addition to the questionnaires conducted after each task, we created a subjective evaluation. We found that all participants found the task to be challenging, but solved them good with both systems given the strict time limit. All participants found the important authors of a topic, regardless of query and system, quickly. They were all able to identify certain key terms, although not all participants necessarily found the same terms. Not all of them managed to identify important publications in the given time. We observed that participants worked a little slower with the prototype than with KOPS. We saw two major reasons for that in our observation: 1) With KOPS, users never looked beyond page three of the result set, shrinking down their result set and 2) users were all very used to working with a browser search, pen & paper and were introduced to a completely novel system on the other hand. From their 30 minutes time, participants usually used the first few minutes to ask about the system and if they could or how to perform certain actions, even though they had received an introduction. But, when using the prototype, almost all participants stated afterwards that "if they were given another task, they could do much more this time". We also noticed that assessing the importance of an author and the area of expertise of a person was especially difficult with KOPS, something participants found much easier in our system.

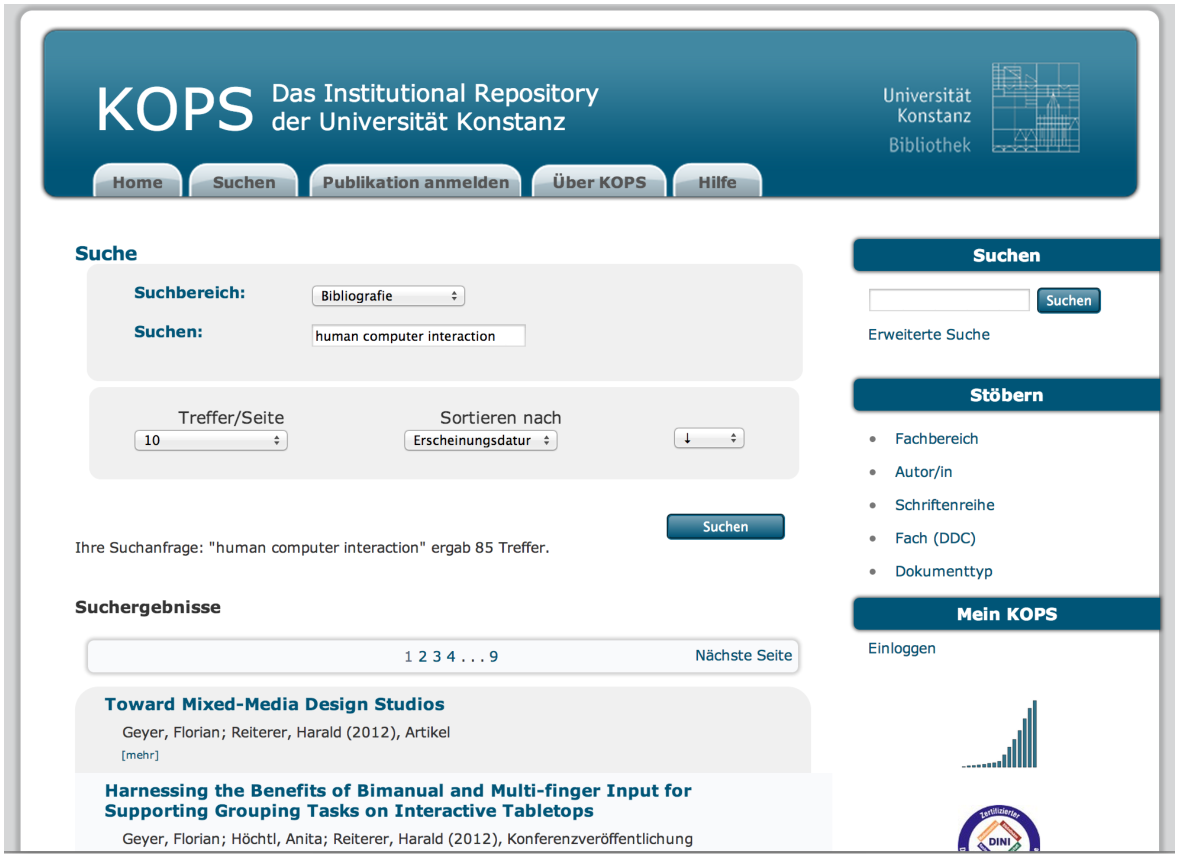
When looking at the structure of the results created (one example per system can be seen in Figure 36 on page 62 and Figure 37 on page 63), we noticed that users tended to work more sequentially when using pen & paper and worked more with spatial arrangement, arrows and connections with the prototype. Users did rarely use size as an indicator for importance, they tended to underline or in some other way mark objects with the pen.

When users were asked about their subjective impressions of the systems, users preferred the prototype in most regards. On a scale from 1 ("is not true") to 7 ("is true") the mean scores per system for "The system helped me efficiently solve the task" was 5.63 (SD = 0.74) for the prototype and 4.71 (SD = 1.8) for KOPS. When asked "The system has supported me well for the task", the mean scores were 5.5 (SD = 1.1) for the prototype and 5.57 (SD = 0.98) for KOPS, slightly in favour of KOPS. While both results are not statistically significant, we still find them interesting, considering users had to deal with a completely novel system when using the prototype. Statistical significance existed when users were answering "The system was valuable for the task". The result was significantly in favour of the prototype ($4.87 > 3.86$; $p < 0.05$). The same is true when users answered "The system was good for the task" ($5 > 3.57$; $p < 0.05$). Users were in favour of KOPS when answering "The system was predictable" ($4.25 < 6.29$; $p < 0.01$). On the other hand, users were clearly in favour of the prototype in any question regarding fun of usage, perceived innovativeness or attractiveness. For example, answering "The system was sympathetic" the result was highly significantly in favour of the prototype ($5.12 > 2.29$; $p < 0.001$). The same is true for "The systems is attractive" ($5.25 > 2.86$; $p < 0.001$) and "The system is innovative" ($5.63 > 1.86$; $p < 0.001$). When answering "The system was activating" the result was highly significantly in favour of the prototype ($5 > 1.86$; $p < 0.001$). The same is true for "The system was exciting" ($5.88 > 2.71$; $p < 0.001$).

Discussion Some of the results of the study were expected, others surprised us. For example we expected users to conduct that KOPS is more predictable and in conformance with expectation than the prototype. All users of the study were very experienced with browser search systems and immediately understood KOPS. Using the prototype, the system and style of work was completely novel to them and it usually took a few minutes for users to get used to this kind of work. For the same reasons, we also expected users to take longer to perform the tasks when working with the prototype. We found it highly interesting that users considered the work with the prototype more efficient than with KOPS and that users found our system more understandable. Considering the novelty of the topic-based approach, we expected users to have more problems in understanding how the system presents search results and how they could explore documents, which proved to be not true. We found it very supporting that users preferred the prototype in every regard considering fun, innovation and attractiveness. This is backed up by users repeatedly expressing opinions like "this way researching is much more fun", "fascinating", "this is awesome", ..., even without being specifically asked for their opinion on the system. Additionally, we observed users being much more interested in the task when using our system, which is backed up by them answering questions for "activeness" or "excitement" clearly in favour of the prototype. Users were also observed to be a lot more playful when using the prototype, trying out different strategies and exploring certain authors or documents just because they had fun in doing it. This could support serendipity findings when using our prototype, something we did not aim for or expect. Two users even drew flowers on their Personal Landscape.

Observed Problems While users had almost no problems understanding the topic-based approach and exploring the data space, we observed some problems in understanding the Personal Landscape. Users sometimes were not sure which objects were already on the Personal Landscape and multiple users expressed their wish to be able to quickly switch back and forth between both landscapes and having a visual link between an object on the Personal Landscape and its source on the search result landscape. One user also tried to annotate on the search result landscape. This leads us to believe that either a merge of both landscapes or a more distinct separation of both is desirable. We also observed that users hesitated more in writing digitally than they did with pen & paper, although users expressed a high amount of fun in this feature. Most users immediately started writing on paper but thought much more about what to write down when using the prototype. Additional visual cues, marking the Personal Landscape as a place for thoughts, could help solve this, for example a paper-like background and a generally less polished look. Also, the possibility to erase notes drawn on the Personal Landscape is necessary to support this process.

Additionally, we observed that users rarely used the co-authors and related documents in the author detail view. A first step in solving this would be to make them more useful by allowing users to tap them in order to open the detail view for the co-author or the details for the related document. Secondly, further visual integration of those seem desirable, for example a link between co-authors and the authors on the search result landscape, if present there. An explanation why a certain document is considered "related" might be useful as well.



Suchbereich: Bibliografie

Suchen: human computer interaction

Treffer/Seite: 10

Sortieren nach: Erscheinungsdatum ↓

Ihre Suchanfrage: "human computer interaction" ergab 85 Treffer.

Suchergebnisse

Toward Mixed-Media Design Studios
Geyer, Florian; Reiterer, Harald (2012), Artikel

Harnessing the Benefits of Bimanual and Multi-finger Input for Supporting Grouping Tasks on Interactive Tabletops
Geyer, Florian; Höchtl, Anita; Reiterer, Harald (2012), Konferenzveröffentlichung

Figure 35: The KOPS search system of the University of Konstanz with the result page for the query "human computer interaction". At the top, users can adjust the number of hits per page and sort by relevance or year. Then, a textual list of results follows. Source: [57]

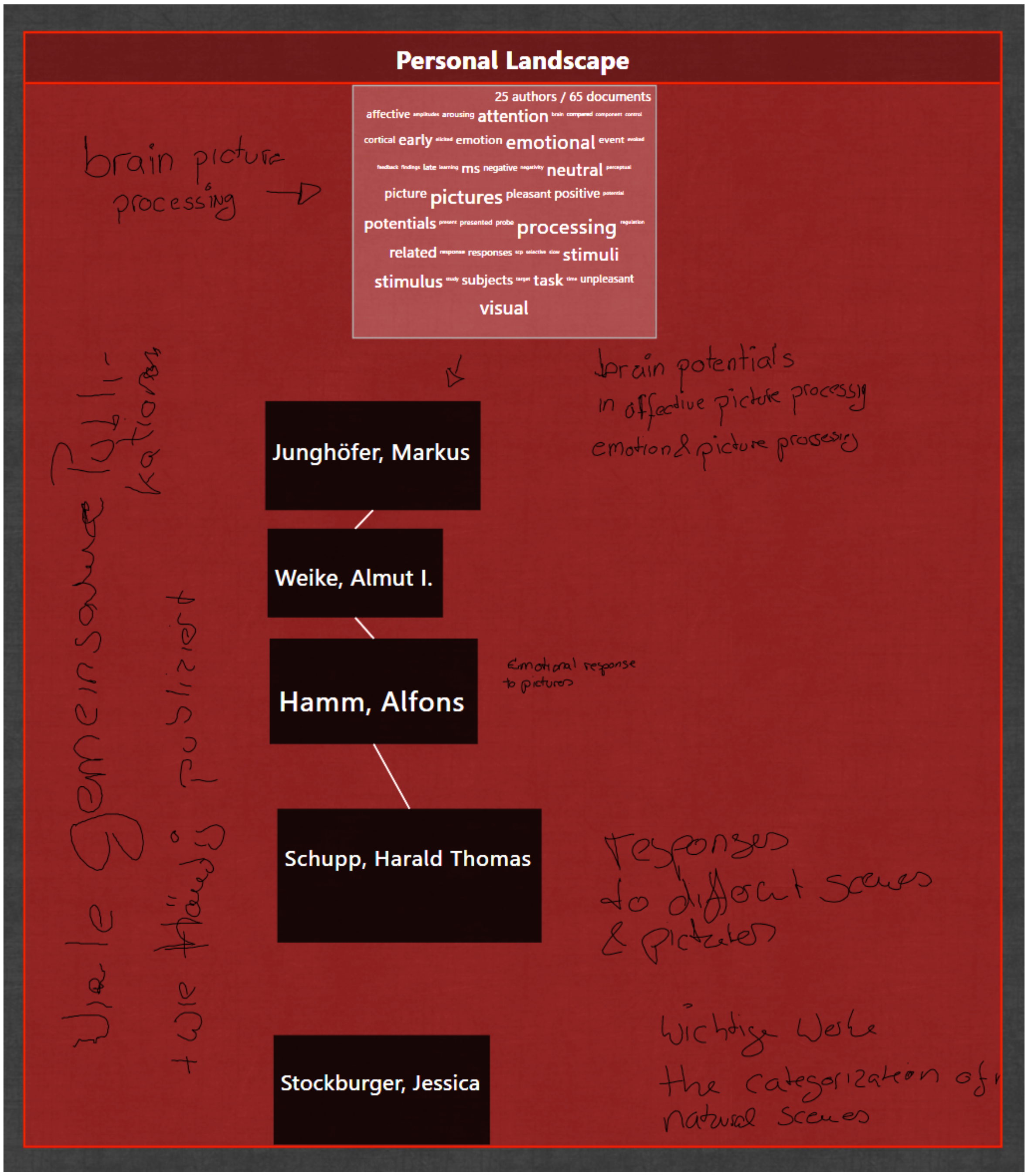
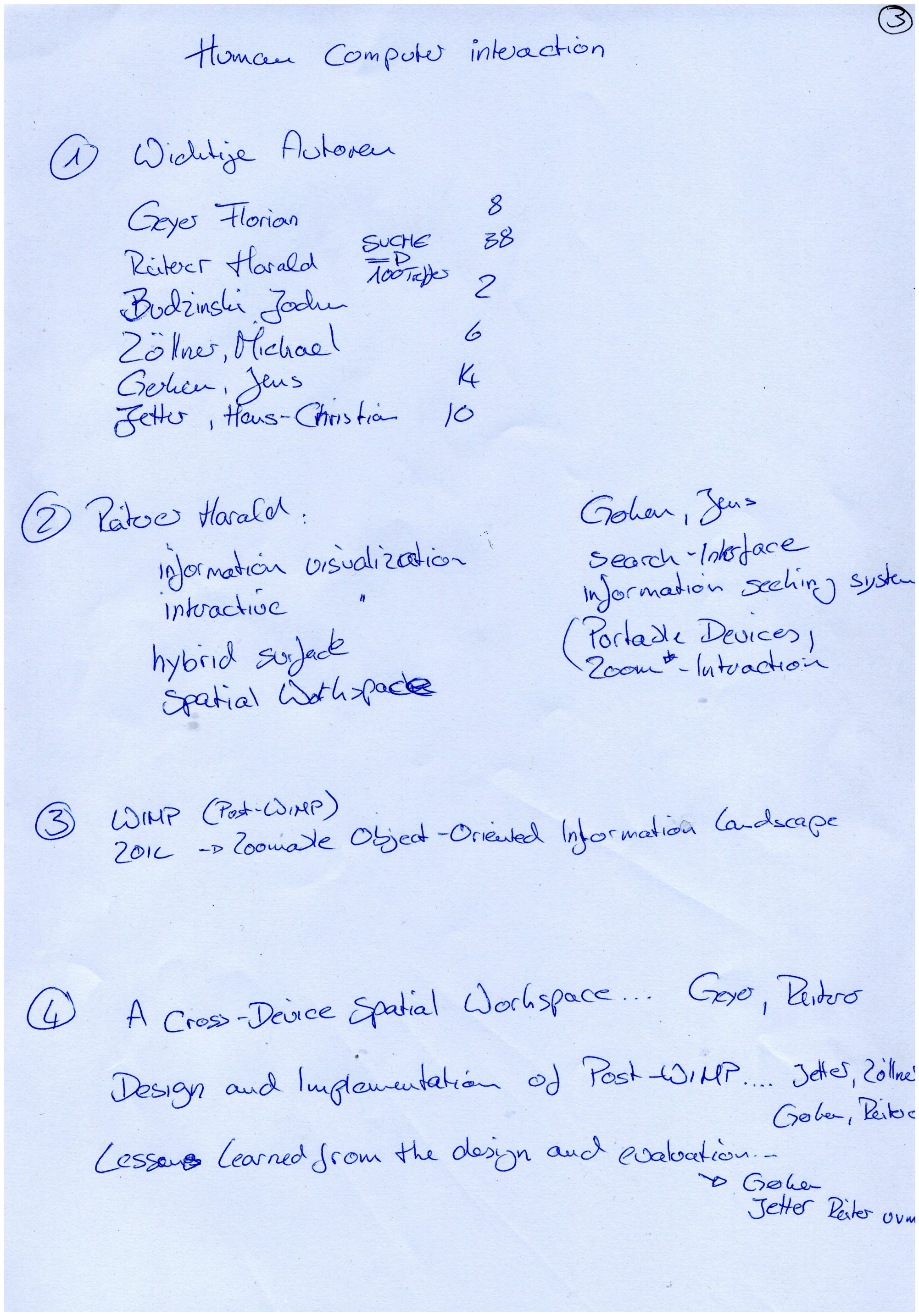

Personal Landscape
25 authors / 65 documents
affective amplitudes arousing attention brain compared component control
cortical early elicited emotion emotional event evoked
feedback findings late learning ms negative negativity neutral perceptual
picture pictures pleasant positive potential
potentials present presented probe processing regulation
related response responses scp selective slow stimuli
stimulus study subjects target task time unpleasant
visual
brain picture processing
brain potentials in affective picture processing
emotion & picture processing
Junghöfer, Markus
Weike, Almut I.
Hamm, Alfons
Schupp, Harald Thomas
Stockburger, Jessica
Emotional response to pictures
responses to different scenes & pictures
Wichtige Werke
the categorization of natural scenes
Viele gemeinsame Publikationen
+ wie häufig publiziert

Figure 36: One user study result created with our prototype. It can be seen that objects and annotations are blended together to create the Personal Landscape. The landscape seems more freely drawn than the result from KOPS (Figure 37).



③

Human Computer interaction

① Wichtige Autoren

Geyer Florian		8
Reiterer Harald	SUCHE => 100 Treffer	38
Budzinski, Jochen		2
Zöllner, Michael		6
Gerken, Jens		14
Jetter, Hans-Christian		10

② Reiterer Harald:
information visualization
interactive "
hybrid surface
spatial Workspace

Gerken, Jens
search-Interface
information seeking system
(Portable Devices,
Zoom*-Interaction

③ WIMP (Post-WIMP)
ZOIL -> Zoomable Object-Oriented Information Landscape

④ A Cross-Device Spatial Workspace ... Geyer, Reiterer

Design and Implementation of Post-WIMP.... Jetter, Zöllne
Gerken, Reite

Lessons learned from the design and evaluation...
-> Gerken
Jetter Reiter usw

Figure 37: One user study result created with KOPS. It can be seen that the result appears to be more linear than in Figure 36.

CHAPTER 6

CONCLUSION

In this thesis, we presented and examined past and present work in the area of visual information seeking systems. We looked at the concepts behind different visual information seeking systems (VISS) and how those systems evolved over the last two decades. We also identified valuable guidelines and concepts for the creation of such systems. Furthermore, we presented work in the area of theoretical search and the search process. We saw different models of search, defined our understanding of the search process and which parts of it are important in a library context. We also defined the process of sense-making, a part of the search process. Furthermore, we looked at the project *Blended Library* at the University of Konstanz and what support it offered for workers seeking for knowledge. We identified a lack of support for browsing and sense-making and then went on to create a visual information seeking system prototype of our own to support those tasks.

For our own VISS prototype, we wanted to create an integrated solution that supported different phases of search and aid the knowledge worker while integrating our prototype into the existing concept of the *Blended Library*. First, we looked at the phase of *query formulation*, or expressing the information need. We made use of another system available in the *Blended Library*, *FacetSearch++*, to support this. *FacetSearch++* offers visual query formulation to the user that allows even novel users to build complex boolean queries. Because we did want the application to support query formulation as a standalone system, for example when it was used on a mobile device, we additionally implemented an alternative method of query formulation: A single-input, cross-metadata search that additionally searched documents by topic which were determined using the topic-modelling algorithm LDA.

We also implemented support for a visual, topic-based approach to browsing. After searching by either using *FacetSearch++* or the single-input search, the resulting documents are grouped by authors and those authors are then assigned to topics by using the LDA algorithm. The user is then presented with a two-dimensional information landscape created with the ZOIL framework, where objects have a defined size and position and are represented visually. At first, topics are displayed on the landscape and each shows the three most common words of documents belonging to that topic. Additionally, topics that revolve around the same theme are spatially grouped together. The user can use this abstract view on the data space to decide which topics are of interest to him and which are not. The user can then use multi-touch interaction to pan and zoom. We are using the concept of *semantic zoom* to allow objects to display different content based on the amount of space they have available on the screen. When the user zooms into topics of interest, they first reveal their fifty most common words. If zoomed in even further, the authors that wrote documents in that topic will be revealed. Those authors are visualised by the years they published in that topic on the x-axis and by the importance in the topic as determined by the LDA algorithm on the y-axis. Additionally, a line chart for each author shows in detail how much the author published each year in the topic. The user can tap an author to open a popup with details about the person. This includes documents written in the topic, all documents, related documents and common co-authors. Documents can be tapped to read their abstract and see documents related to them.

Furthermore, the author details include a custom-built glyph called *ChromeViz*. When using *FacetSearch++* the user has the possibility to put up to three tokens simultaneously on the table and by that send up to three queries to our prototype at the same time. ChromeViz is a glyph that gives information about how many documents an author wrote in each of those three result sets, and if documents are in more than one result set. For that, ChromeViz constructs a modified pie chart, where a "bleeding effect", an overlap between segments of the chart, is used to indicate overlap of result sets. While ChromeViz does not give the user exact data, it allows for a quick evaluation and comparison of the authors documents in regard to the different result sets.

Finally, we added support for the sense-making step in our prototype. An additional landscape, the Personal Landscape, is used for that. Using drag gestures user can copy interesting objects from the result sets onto the Personal Landscape. Even when a new search is conducted, personalised objects are permanent. The user has the same abilities as on the search result landscape, zooming and spatially arranging objects. Additionally, a touch-pen allows the user to perform annotations to note down thought and findings on the objects. Drawing a line between two objects links them, indicating relationship. Furthermore, the user can use the pen to free-draw anywhere on the Personal Landscape. This allows him to write, draw, create mind-maps or otherwise note his concepts.

In the end, we allowed the user to get from the query formulation and then a very abstract overview over the data space to individual documents and right into sense-making within a single application context. The user does not have to switch between windows, tabs and there is no additional UI to switch between the different phases of the search process. Also, the user can naturally explore the data space by panning and zooming, while zooming reveals more details by using *semantic zoom*. This allows the user to filter the data space implicitly, without the need for a dedicated UI, by zooming into the parts that are of interest to the research.

We put parts of those concepts to the test in a user study. To keep the number of influencing factors as small as possible, we focused on evaluating the exploration of a topic and the sense-making. Users had to get a first, broad overview over a completely unknown topic within 30 minutes. Even though our topic-based approach was new to them and they had to use a completely novel system, they achieved similar results as with the default web based search system of the University of Konstanz, KOPS. Additionally, a subjective evaluation showed that, while users favoured KOPS in regard to predictability and the tried-and-true web search they knew, our system was considered more valuable and in every regard more innovative and fun to work with. Additionally, users seemed more motivated to solve the task with our system and said afterwards the system was more exciting and activating. Although the study also showed some problems, for example users did not fully understand the concept of a separate Personal Landscape and hesitated more to write down notes digitally, this shows that our concepts work well and are well accepted by users.

6.1 Future Work

With a prototype of this scope, it is natural that there are still things that could be added in the future. This section will name a few features that we could not explore in the scope of a bachelor project. These features could be added in future work on this concept to improve the concept and the user experience with the system.

6.1.1 Extended query formulation

In our own approach to query formulation, we used a single-input search that searches across all metadata and across topics. Furthermore, we implemented implicit filtering by browsing the data space. Because of the scope of this work, we focused on our contribution to the query formulation step and did not re-implement existing solutions. In the future, some of those solutions could make a useful addition to our search, for example query preview, query completion or support for boolean queries.

6.1.2 Topic-Modelling

As we talked about in Section 4.5.1 on page 40, there are other topic-modelling algorithms available than the LDA. We saw *SOMLib* by Rauber and Merkl[63] and the suffix-tree clustering by Zamir and Etzioni[87], to name just two others. The different ways of topic-modelling should be evaluated in regard to applicability and computational complexity to see which of the available topic-modelling algorithms is best suited for this system. This is especially interesting, because performing the topic-modelling online (during runtime, only for the candidate results) could be additionally useful.

6.1.3 Author positioning

In Listing 3 on page 46 we showed the algorithm that positions authors inside a topic. We also talked about that this algorithm tries to position authors by their topic affiliation on the y-axis, but is not stable in doing so. Developing an alternative, stable algorithm is desirable, as a guaranteed ordering of authors on the y axis would help the user in further evaluating authors.

6.1.4 Extended search history

We talked about our ways of expressing the information need and searching in Section 4.4 on page 35. There, we also talked about the search sidebar and the search history. We are confident that it would be useful if this history would be further extended, allowing users to switch between search results instantly and having the exact same search result landscape they had when they left the search, including spatial arrangement or sizing of objects. This could help the user in comparing searches and reconstructing the train of thought after a long search process.

6.1.5 Personal Landscape

As we saw as a result of our user study in Chapter 5 on page 58, users were sometimes confused with the concept of a separate Personal Landscape. We think even small additions to the landscape, like a quick way to jump between the Personal Landscape and the search results or a more note-taking like background for the Personal Landscape, for example a paper-related background image, could help solve those issues. In the future, it could be possible for the Personal Landscape to be merged with the search results landscape to form a single landscape.

6.1.6 Additional author data

Our user study in Chapter 5 on page 58 also showed us that users rarely used the additional author metadata, co-authors and related documents. We already proposed some possible solutions for this: One solution could be to make them more useful by adding interaction, for example opening information about a related documents when tapped. Another solution could be to link co-authors to the topics on the search result landscape where they are contained. An explanation why a certain document is considered "related" might be useful as well.

6.1.7 Collaboration and Sharing

This thesis did not focus on the topics of collaboration and sharing. Adding features to collaboratively search, browse and make sense and to share findings, searches or products would bring entirely new facets to this concept. Additionally, supporting collaboration and sharing would further integrate the phases of the search models we discovered, for example the "Relate" and "Donate" phases mentioned in Shneiderman's Framework for Mega-Creativity (discussed in Section 3.1 on page 20).

CHAPTER 7

THANKS

In the end, there are some personal “Thank You”s I would like to express. Because of the personal nature of those expressions of thanks, I will write them down in my native language of German.

Vor allem und allen anderen, und aus tiefstem Herzen, möchte ich meinen Eltern danken! Ohne ihre ständige Unterstützung in allen Bereichen meines Lebens wäre mein Studium, diese Arbeit und so vieles andere nie zustande gekommen. Auch in schwierigen Phasen hatten sie immer Unterstützung und Verständnis. Danke!

Vielen Dank auch an alle Freunde, die mich während dieser Zeit mit Rat, Aufheiterung und Ablenkung begleitet haben und auch in Stresszeiten Verständnis aufgebracht haben.

Des weiteren möchte ich auch besonderen Dank an Roman Rädle, meinem Betreuer während der gesamten Entwicklung dieser Arbeit, aussprechen. Trotz vielen weiteren Verpflichtungen und einem engen Zeitplan hatte er immer Zeit, Unterstützung, fachlichen Rat und gute Anstösse für mich, was ich sehr zu schätzen weiss.

Zusätzlich gilt mein Dank allen weiteren Mitarbeitern des HCI-Lehrstuhls an der Universität Konstanz, die mir in welcher Form auch immer während der Entwicklung dieser Arbeit geholfen haben. Und natürlich Prof. Reiterer für die Möglichkeit, diese Arbeit an seinem Lehrstuhl zu entwickeln.

Als letztes noch Dank an die Teilnehmer der Benutzerstudie, die ihre Zeit und Mühe der Forschung gewidmet haben.

Danke!

CHAPTER 8

BIBLIOGRAPHY

[1] Anoto Group AB. `http://www.anoto.com`, 08.06.2013.

[2] Christopher Ahlberg and Ben Shneiderman. Visual information seeking: tight coupling of dynamic query filters with starfield displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '94, pages 313–317, New York, NY, USA, 1994. ACM.

[3] Christopher Ahlberg and Ben Shneiderman. Visual information seeking using the filmfinder. In *Conference Companion on Human Factors in Computing Systems*, CHI '94, pages 433–434, New York, NY, USA, 1994. ACM.

[4] Apple. `http://support.apple.com/kb/ht2531`, 07.03.2013.

[5] Arvind Arasu, Junghoo Cho, Hector Garcia-Molina, Andreas Paepcke, and Sriram Raghavan. Searching the web. *ACM Trans. Internet Technol.*, 1(1):2–43, August 2001.

[6] Human-Computer-Interaction Group at the University of Konstanz. `http://hci.uni-konstanz.de/index.php?a=research&b=projects&c=8609071`, 13.08.2013.

[7] Human-Computer-Interaction Group at the University of Konstanz. `http://www.youtube.com/watch?v=099ZiJTr1pA`, 12.07.2013.

[8] Human-Computer-Interaction Group at the University of Konstanz. Video containing some features of the Media Seminar Room: `http://www.youtube.com/watch?v=Np1ODKU48do`.

[9] Human-Computer-Interaction Group at the University of Konstanz. `http://www.youtube.com/watch?v=giDF9lKhCLc`, 08.06.2013.

[10] T. Baube. Interactive reading of digital documents in the mobile context. Universität Konstanz, 2013.

[11] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[12] Christine L. Borgman. Why are online catalogs hard to use? Lessons learned from information-retrieval studies. *J. Am. Soc. Inf. Sci.*, 37(6):387–400, January 1986.

[13] Christine L. Borgman. Why are online catalogs still hard to use? *Journal of the American Society for Information Science*, 47(7):493–503, 1996.

[14] Thorsten Buering and Harald Reiterer. Zuiscat - querying and visualizing information spaces on personal digital assistants. In *MobileHCI 2005: Human Computer Interaction with Mobile Devices and Services, Proceedings of the MobileHCI 2005*. ACM Press, Sep 2005.

[15] Stuart K. Card, George G. Robertson, and William York. The webbook and the web forager: video use scenarios for a world-wide web information workspace. In *Conference companion on Human factors in computing systems: common ground*, CHI '96, pages 416–417, New York, NY, USA, 1996. ACM.

[16] Richard Chimera and Ben Shneiderman. An exploratory evaluation of three interfaces for browsing large hierarchical tables of contents. *ACM Trans. Inf. Syst.*, 12(4):383–406, October 1994.

[17] DISPLAX. `http://www.displax.com/en/products/products/skin-multitouch.html`, 14.08.2013.

[18] Offer Drori. Displaying search results in textual databases. *SIGCHI Bull.*, 32(1):73–77, January 2000.

[19] Dennis E. Egan, Joel R. Remde, Thomas K. Landauer, Carol C. Lochbaum, and L. M. Gomez. Behavioral evaluation and analysis of a hypertext browser. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '89, pages 205–210, New York, NY, USA, 1989. ACM.

[20] Brynn M. Evans and Ed H. Chi. Towards a model of understanding social search. In *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, CSCW '08, pages 485–494, New York, NY, USA, 2008. ACM.

[21] Brynn M. Evans and Ed H. Chi. An elaborated model of social search. *Inf. Process. Manage.*, 46(6):656–678, November 2010.

[22] G W. Furnas, T K. Landauer, L M. Gomez, and S T. Dumais. Human factors in computer systems. chapter Statistical semantics: analysis of the potential performance of keyword information systems, pages 187–242. Ablex Publishing Corp., Norwood, NJ, USA, 1984.

[23] Christoph Gebhard. `http://www.youtube.com/watch?v=0VCYwEmNdws`, 14.08.2013.

[24] Christoph Gebhard. Integrative workplace - employing reality-based interaction to join digital and analog media at a workplace. Universität Konstanz, 2013.

[25] Jens Gerken, Mathias Heilig, Hans-Christian Jetter, Sebastian Rexhausen, Mischa Demarmels, Werner A. K&#x00f6;nig, and Harald Reiterer. Lessons learned from the design and evaluation of visual information-seeking systems. *Int. J. Digit. Libr.*, 10(2-3):49–66, December 2009.

[26] BaseX GmbH. `http://basex.org`, 11.08.2013.

[27] Christian Grün, Jens Gerken, Hans-Christian Jetter, Werner A. König, and Harald Reiterer. Mediovis - a user-centred library metadata browser. In *ECDL 2005: Proceedings of the 9th European Conference on Research and Advanced Technology for Digital Libraries 2005*, pages 174–185. Springer Verlag, Sep 2005.

[28] S. Göbel, J. Haist, Frank Müller, and Harald Reiterer. Invisip: Usage of information visualization techniques to access geospatial data archives. In *DEXA 2002: Proceedings of the 13th International Conference on Database and Expert Systems Applications*, pages 371–380. Springer, Sep 2002.

[29] Mathias Heilig and Harald Reiterer. Mediovis 2.0 - neue interaktionsformen für die bibliothek der zukunft. 2010.

[30] Mathias Heilig, Roman Rädle, and Harald Reiterer. *Die Blended Library: Benutzerorientierte Verschmelzung von virtuellen und realen Bibliotheksdiensten.* De Gruyter, Oct 2011.

[31] Nadine Höchstötter and Dirk Lewandowski. What users see - structures in search engine results pages. *Inf. Sci.*, 179(12):1796–1812, May 2009.

[32] Google Inc. `http://www.google.com`, 04.08.2013.

[33] Google Inc. `http://support.google.com/webmasters/bin/answer.py?hl=en&answer=70897`, 07.03.2013.

[34] Google Inc. `https://www.google.de/search?q=whel&aq=f&oq=whel`, 07.03.2013.

[35] Google Inc. `http://www.google.com/chrome`, 28.07.2013.

[36] Hans-Christian Jetter, Jens Gerken, Werner A. König, Christian Grün, and Harald Reiterer. Hypergrid - accessing complex information spaces. In *HCI UK 2005: People and Computers XIX - The Bigger Picture, Proceedings of the 19th British HCI Group Annual Conference 2005*, pages 349–364. Springer Verlag, Sep 2005.

[37] Hans-Christian Jetter, Jens Gerken, Michael Zöllner, Harald Reiterer, and Natasa Milic-Frayling. Materializing the query with facet-streams – a hybrid surface for collaborative search on tabletops. In *CHI'11: Proceedings of the 29th international conference on Human factors in computing systems*, pages 3013–3022. ACM Press, May 2011.

[38] Hans-Christian Jetter, Werner A. König, Jens Gerken, and Harald Reiterer. Zoil - a cross-platform user interface paradigm for personal information management. In *CHI 2008 Workshop - The Disappearing Desktop: Personal Information Management 2008*, Apr 2008.

[39] Hans-Christian Jetter, Michael Zöllner, Jens Gerken, and Harald Reiterer. Design and implementation of post-wimp distributed user interfaces with zoil. *International Journal of Human-Computer Interaction*, 28(11 (Special Issue: Distributed User Interfaces)):737–747, Sep 2012.

[40] Peter Klein, Harald Reiterer, Frank Müller, and Tobias Limbach. Metadata visualization with vismeb. In *IV03: Proceedings of the 7th International Conference on Information Visualisation 2003*, pages 600–605. IEEE Computer Society, Jul 2003.

[41] Eike Kleiner, Roman Rädle, and Harald Reiterer. Blended shelf: Reality-based presentation and exploration of library collections. In *In Proc. CHI'13: Proceedings of the 31st international conference extended abstracts on Human factors in computing systems, Work-In-Progress Session.* ACM Press, Apr 2013.

[42] Eike Kleiner and Benjamin Schaefer. Augmented shelf: Digital enrichment of library shelves. In *In Proceedings of Mensch und Computer 2012 - Interaktiv Informiert - Konstanz, Germany, inter—aktion, Demo Session.* Oldenbourg, Sep 2012.

[43] Tobias Limbach, Frank Müller, Peter Klein, Harald Reiterer, and Maximilian Eibl. Visualization of metadata using the supertable+scatterplot. In *Schriften zur Informationswissenschaft: Information und Mobilität, 8. Internationales Symposium für Informationswissenschaft*, pages 147–163. UKV Universitätsverlag Konstanz, Sep 2002.

[44] Tobias Limbach, Harald Reiterer, Peter Klein, and Frank Müller. Vismeb: A visual metadata browser. In *Interact 2003: Proceedings of the Ninth IFIP TC13 International Conference on Human-Computer Interaction*, pages 993–996. IOS Press, Sep 2003.

[45] T. M. Mann and H. Reiterer. Evaluation of different visualizations of web search results. In *Proceedings of the 11th International Workshop on Database and Expert Systems Applications*, DEXA '00, pages 586–, Washington, DC, USA, 2000. IEEE Computer Society.

[46] Andrew Kachites McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

[47] Panagiotis Takis Metaxas. On the evolution of search engine rankings. In *In In the Proceedings of the 2009 WEBIST Conference.*

[48] Microsoft. `http://www.microsoft.com/en-us/pixelsense/applicationpages10.aspx`, 13.08.2013.

[49] Microsoft. `http://www.microsoft.com/en-us/pixelsense/WhatsNew.aspx`, 08.06.2013.

[50] Micrsofot. `http://academic.research.microsoft.com/`, 10.08.2013.

[51] G. Mußler. *Retrieving business information from the WWW.* 2002.

[52] Frank Müller, Peter Klein, Tobias Limbach, and Harald Reiterer. Visualization and interaction techniques of the visual metadata browser vismeb. In *I-Know 03: Proceedings of 3rd International Conference on Knowledge Management*, pages 89–95. Knowcenter, Jul 2003.

[53] Chris North and Ben Shneiderman. A taxonomy of multiple window coordinations. Technical report, 1997.

[54] Chris North and Ben Shneiderman. Snap-together visualization: Coordinating multiple views to explore information. Technical report, University of Maryland Computer Science, 1999.

[55] Rabia Nuray-Turan, Dmitri V. Kalashnikov, and Sharad Mehrotra. Exploiting web querying for web people search. *ACM Trans. Database Syst.*, 37(1):7:1–7:41, March 2012.

[56] University of Konstanz. `http://www.ub.uni-konstanz.de/home/`, 08.06.2013.

[57] University of Konstanz. `http://kops.ub.uni-konstanz.de`, 17.08.2013.

[58] Ken Perlin and David Fox. Pad: an alternative approach to the computer interface. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '93, pages 57–64, New York, NY, USA, 1993. ACM.

[59] P. Pirolli and S. Card. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of International Conference on Intelligence Analysis*, 2005.

[60] Annabel Pollock and Andrew Hockley. What"s wrong with internet searching. Technical report, 1997.

[61] Jef Raskin. *The humane interface: new directions for designing interactive systems.* ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 2000.

[62] Andreas Rauber. Labelsom: On the labeling of self-organizing maps. In *In Proc. International Joint Conference on Neural Networks*, pages 1–6, 1999.

[63] Andreas Rauber and Dieter Merkl. Somlib: a digital library system based on neural networks. In *Proceedings of the fourth ACM conference on Digital libraries*, DL '99, pages 240–241, New York, NY, USA, 1999. ACM.

[64] Harald Reiterer, Mathias Heilig, and Sebastian Rexhausen. Mediovis 2.0 - a novel user interface for seeking audio-visual media libraries. In *Workshop AudioVisuelle Medien WAM 2009.* Technische Universität Leipzig, Jun 2009.

[65] Harald Reiterer, Hans-Christian Jetter, Werner A. König, Jens Gerken, and Christian Grün. Zoomtechniken zur exploration komplexer informationsräume am beispiel hypergrid. In *Mensch & Computer 2005: Kunst und Wissenschaft - Grenzüberschreitungen der interaktiven ART*, pages 143–153. Oldenbourg Verlag, Sep 2005.

[66] Harald Reiterer, Tobias Limbach, Peter Klein, Frank Müller, and Hans-Christian Jetter. Ein visueller metadaten browser für die explorative erkundung großer datenmengen. In *Mensch und Computer 2003 - Interaktion in Bewegung*, pages 165–176. B.G. Teubner, Sep 2003.

[67] Harald Reiterer and Thomas M. Mann. Visual information retrieval for the web, 2000.

[68] Harald Reiterer, Gabriela Muer, Thomas M. Mann, and Siegfried H. Insyder - an information assistant for business intelligence. In *Proceedings of the 23 Annual International ACM SIGIR 2000 Conference on Research and Development in Information Retrieval, ACM press*, pages 112–119, 2000.

[69] Harald Reiterer, Gabriele Tullius, and Thomas M. Mann. Insyder: a content-based visual-information-seeking system for the web. *International Journal on Digital Libraries*, 5(1):25–41, Mar 2005.

[70] Gene Roddenberry and Michael Piller. `http://www.imdb.com/title/tt0708785/?ref_=fn_tt_tt_11`, 03.08.2013.

[71] Roman Rädle. Human-Computer-Interaction Group at the University of Konstanz, `http://hci.uni-konstanz.de/?a=research&b=livinglab&lang=en`.

[72] Roman Rädle, Hans-Christian Jetter, and Harald Reiterer. Twistersearch: A distributed user interface for collaborative web search. In *Proceedings of the 2nd Workshop on Distributed User Interfaces: Collaboration and Usability (In conjunction with CHI 2012 Conference)*, pages 1–4. University of Castilla-La Mancha, Spain, May 2012.

[73] Roman Rädle, Hans-Christian Jetter, and Harald Reiterer. Twistersearch: Supporting social search with tabletop and mobile displays. In *In Proceedings of Mensch und Computer 2012 - Interaktiv Informiert - Konstanz, Germany, inter—aktion, Demo Session.* Oldenbourg, Sep 2012.

[74] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of the 1996 IEEE Symposium on Visual Languages*, VL '96, pages 336–, Washington, DC, USA, 1996. IEEE Computer Society.

[75] Ben Shneiderman. Codex, memex, genex: the pursuit of transformational technologies. In *CHI 98 Cconference Summary on Human Factors in Computing Systems*, CHI '98, pages 98–99, New York, NY, USA, 1998. ACM.

[76] Ben Shneiderman. Creating creativity: user interfaces for supporting innovation. *ACM Trans. Comput.-Hum. Interact.*, 7(1):114–138, March 2000.

[77] Ben Shneiderman. Leonardo's laptop: human needs and the new computing technologies. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, CIKM '05, pages 1–1, New York, NY, USA, 2005. ACM.

[78] Ben Shneiderman, Don Byrd, and W. B Croft. Clarifying search: A user-interface framework for text searches. Technical report, 1997.

[79] Alan Smeaton and Francis Crimmins. Relevance feedback and query expansion for searching the web: A model for searching a digital library. In *In: Research and Advanced Technology for Digital Libraries, First European Conference ECDL'97, C Peters and C Thanos (Eds.) Springer LNCS*, pages 99–112, 1997.

[80] Karen Sparck Jones. Document retrieval systems. chapter A statistical interpretation of term specificity and its application in retrieval, pages 132–142. Taylor Graham Publishing, London, UK, UK, 1988.

[81] Amanda Spink, Bernard J. Jansen, Dietmar Wolfram, and Tefko Saracevic. From e-sex to e-commerce: Web search changes. *Computer*, 35(3):107–109, March 2002.

[82] Craig Tashman. Liquidtext: active reading through multi-touch document manipulation. In *Proceedings of the 28th of the international conference extended abstracts on Human factors in computing systems*, CHI EA '10, pages 2959–2962, New York, NY, USA, 2010. ACM.

[83] Craig S. Tashman and W. Keith Edwards. Active reading and its discontents: the situations, problems and ideas of readers. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 2927–2936, New York, NY, USA, 2011. ACM.

[84] Ellen M. Voorhees. Query expansion using lexical-semantic relations. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '94, pages 61–69, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[85] WayBackMachine. `http://web.archive.org/web/19961017235908/http://www2.yahoo.com`, 07.03.2013.

[86] Christopher Williamson and Ben Shneiderman. The dynamic homefinder: evaluating dynamic queries in a real-estate information exploration system. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '92, pages 338–346, New York, NY, USA, 1992. ACM.

[87] Oren Zamir and Oren Etzioni. Web document clustering: a feasibility demonstration. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '98, pages 46–54, New York, NY, USA, 1998. ACM.

[88] Justin Zobel and Alistair Moffat. Inverted files for text search engines. *ACM Comput. Surv.*, 38(2), July 2006.

[89] Michael Zöllner, Hans-Christian Jetter, and Harald Reiterer. *ZOIL: A Design Paradigm and Software Framework for Post-WIMP Distributed User Interfaces*, chapter 10, pages 87–94. Human–Computer Interaction Series. Springer-Verlag, Dec 2011.

[90] Michael Zöllner, Hans-Christian Jetter, and Harald Reiterer. Zoil: A design paradigm and software framework for post-wimp distributed user interfaces. In *1st Workshop on Distributed User Interfaces 2011 (a CHI 2011 Workshop)*, pages 37–40. University of Castilla-La Mancha, Spain, May 2011.

Appendices

APPENDIX A

EXAMPLE XML OF A DOCUMENT IN THE BASEX DATABASE

The following listing contains an example document (”Medium”) as it occurs in the BaseX database. The contents of `<Title_Related>` and `<Fulltext>` were shortened, which is marked by “`(...)`”. `<Fulltext>` contains the content of the document if it is available.

```
<Medium mv_id="1317362" bib_id="260125822">
  <Title_Related>
    <Rdium>
      <Id>281148201</Id>
      <Value>0.28867513</Value>
    </Rdium>
    <Rdium>
      <Id>280675496</Id>
      <Value>0.25</Value>
    </Rdium>
    <Rdium>
      <Id>280366647</Id>
      <Value>0.2236068</Value>
    </Rdium>
    <Rdium>
      <Id>27171753X</Id>
      <Value>0.20412414</Value>
    </Rdium>
    <Rdium>
      <Id>286813823</Id>
      <Value>0.18898225</Value>
    </Rdium>

     (...)
  </Title_Related>
  <Authors_Related>
    <Rdium>
      <Id>279220731</Id>
      <Value>0.25</Value>
    </Rdium>
    <Rdium>
      <Id>25985560X</Id>
      <Value>0.0</Value>
    </Rdium>
  </Authors_Related>
  <Abstract_Related>
    <Rdium>
```

```
      <Id>265622433</Id>
      <Value>0.26358542</Value>
    </Rdium>
    <Rdium>
      <Id>266486886</Id>
      <Value>0.2550179</Value>
    </Rdium>
    <Rdium>
      <Id>26657534X</Id>
      <Value>0.2546428</Value>
    </Rdium>
    <Rdium>
      <Id>266441696</Id>
      <Value>0.2543339</Value>
    </Rdium>
    <Rdium>
      <Id>266483887</Id>
      <Value>0.25288472</Value>
    </Rdium>
    <Rdium>
      <Id>260117137</Id>
      <Value>0.24481438</Value>
    </Rdium>

     (...)
  </Abstract_Related>
  <Abstract>The paper presents design and results of an empirical study on organizational ↩
      learning (OL). OL is conceptualized as a communication based process, changing ↩
      organizationally shared reality constructions. A model is developed, which allows ↩
      the description of organizations as learning systems. It serves as a frame for ↩
      collecting and structuring data on reality constructions and communication relations↩
      . Gathered by interviews in two municipal administrations, data is further processed↩
       into cognitive maps (reality constructions) and networks (communication relations).↩
       Their analysis leads to propositions about the nature of OL processes. It is ↩
      demonstrated that changes of organizationally shared reality constructions (OL) ↩
      originate in self-organizing communication networks. Furthermore, a structural and a↩
       strategic organizational learning mode are distinguished. They differ with respect ↩
      to the information which is processed in the networks, the outcome they produce and ↩
      the logic they follow.</Abstract>
  <Author>Klimecki, Rüdiger G.</Author>
  <Author>Laß leben, Hermann</Author>
  <Title>Modes of organizational learning</Title>
  <Desc>Rüdiger G. Klimecki ; Hermann Laß leben</Desc>
  <Town>Konstanz</Town>
  <Publisher>Bibliothek der Universität Konstanz</Publisher>
  <Year>1998</Year>
  <Format>Online-Ressource</Format>
  <Note>ResearchPaper</Note>
  <Signature>|320</Signature>
  <Language>Deutsch</Language>
  <Type>Publikation</Type>
  <Type>Online-Ressource</Type>
  <Category>Politik/Verwaltungswissenschaft</Category>
  <Url url_desc="Resolving-System">http://nbn-resolving.de/urn:nbn:de:bsz:352-opus-335</↩
      Url>
  <Fulltext name="fulltext">
    <Page name="page" number="1"/>
    <Page name="page" number="2"/>
    <Page name="page" number="3"/>
    <Page name="page" number="4"/>
    <Page name="page" number="5"/>
    <Page name="page" number="6"/>
    <Page name="page" number="7"/>
    <Page name="page" number="8"/>
    <Page name="page" number="9"/>
    <Page name="page" number="10"/>
    <Page name="page" number="11"/>
    <Page name="page" number="12"/>
    <Page name="page" number="13"/>

     (...)
  </Fulltext>
</Medium>
```