Design and Implementation of Post-WIMP Interactive Spaces with the ZOIL Paradigm

Dissertation zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften (Dr. rer. nat.)

vorgelegt von

Hans-Christian Jetter

an der

Universität Konstanz

Mathematisch-Naturwissenschaftliche Sektion

Fachbereich Informatik und Informationswissenschaft

Tag der mündlichen Prüfung: 22. März 2013

1. Referent: Prof. Dr. Harald Reiterer (Universität Konstanz)
2. Referent: Prof. Dr. Andreas Butz (LMU München)
Prüfungsvorsitzender: Prof. Dr. Daniel Keim (Universität Konstanz)

Konstanz, Dezember 2013

Design and Implementation of Post-WIMP Interactive Spaces with the ZOIL Paradigm

Dissertation submitted for the degree of
Doctor of Natural Sciences (Dr. rer. nat.)

Presented by

Hans-Christian Jetter

at the

University of Konstanz

Faculty of Sciences

Department of Computer and Information Science

Date of the oral examination: March 22nd, 2013

1st Supervisor: Prof. Dr. Harald Reiterer (University of Konstanz)
2nd Supervisor: Prof. Dr. Andreas Butz (University of Munich, LMU)
Chief Examiner: Prof. Dr. Daniel Keim (University of Konstanz)

Konstanz, December 2013

Zusammenfassung

Post-WIMP (post-“**W**indows, **I**cons, **M**enu, **P**ointer”) interaktive Räume sind physische Umgebungen oder Räume für die kollaborative Arbeit, die mit Technologien des Ubiquitous Computing angereichert sind. Ihr Zweck ist es, eine computer-unterstützte Kollaboration mehrerer Benutzer zu ermöglichen, die auf einer nahtlosen Benutzung mehrerer Geräte und Bildschirme mittels „natürlicher“ post-WIMP Interaktion basiert.

Diese Dissertation beantwortet die Forschungsfrage, wie Gestalter und Entwickler solcher Umgebungen dabei unterstützt werden können, gebrauchstaugliche interaktive Räume für mehrere Benutzer mit mehreren Geräten zu erschaffen, die eine kollaborative Wissensarbeit ermöglichen.

Zu diesem Zweck werden zunächst Konzepte wie post-WIMP Interaktion, interaktive Räume und Wissensarbeit definiert. Die Arbeit formuliert dann das neue technologische Paradigma *ZOIL* (**Z**oomable **O**bject-Oriented **I**nformation **L**andscape). Dieses *ZOIL Paradigma* ist der Hauptbeitrag dieser Arbeit und besteht aus drei Komponenten:

1.) Die sechs *ZOIL Gestaltungsprinzipien*, welche die Art der Benutzerinteraktion mit ZOIL Benutzungsschnittstellen definieren und Interaktionsdesigner mit „goldenen Regeln“ unterstützen.

2.) Das *ZOIL Software Framework*, das Entwickler während der Implementierung von post-WIMP interaktiven Räumen für die kollaborative Wissensarbeit unterstützt und die praktische Umsetzung der *ZOIL Gestaltungsprinzipien* ermöglicht.

3.) Die vier auf ZOIL basierenden *Beispiel-Prototypen*, die Gestaltern und Entwicklern gleichermaßen als Anschauungsobjekte dienen können.

Jedes der sechs *ZOIL Gestaltungsprinzipien* ist aus existierender Literatur hergeleitet. Dazu wurde Literatur aus den Disziplinen Mensch-Computer Interaktion, Ubiquitous Computing, Informationsvisualisierung, Computerunterstützte Gruppenarbeit, Kognitionswissenschaft, Persönliches Informationsmanagement und Software Engineering herangezogen.

Die Formulierung der Prinzipien wird durch eigene Erkenntnisse während der Erstellung der *ZOIL Beispiel-Prototypen* für verschiedene Anwendungsdomänen (z.B. e-Science, kollaborative Suche, Produktdesign) und durch die Ergebnisse von eigenen Benutzerstudien empirisch validiert und erweitert.

Das neue quelloffene *ZOIL Software Framework* dient dem Zweck der Implementierung von post-WIMP Interaktiven Räumen entsprechend den *ZOIL Gestaltungsprinzipien*. Dieses Software Framework wird anhand seiner Architektur und Design Patterns vorgestellt und mit Hinblick auf seine Gebrauchstauglichkeit und seinen praktischen Nutzen für Entwickler im Rahmen einer API-Gebrauchstauglichkeits-Studie evaluiert.

Abstract

Post-WIMP (post-“**W**indows, **I**cons, **M**enu, **P**ointer”) interactive spaces are physical environments or rooms for collaborative work that are augmented with ubiquitous computing technology. Their purpose is to enable a computer-supported collaboration of multiple co-located users that is based on a seamless use of and “natural” interaction with multiple devices and displays.

This thesis answers the research question how the designers and developers of such ubiquitous computing environments can be supported to create more usable multi-user and multi-device post-WIMP interactive spaces for co-located collaborative knowledge work.

For this purpose, it first defines concepts such as post-WIMP interaction, interactive spaces, and knowledge work. Then it formulates the novel *ZOIL* (**Z**oomable **O**bject-Oriented **I**nformation **L**andscape) technological paradigm. The *ZOIL paradigm* is the main contribution of this thesis and consists of three components:

1.) The six *ZOIL design principles* that define ZOIL’s interaction style and provide “golden rules” to support interaction designers.

2.) The *ZOIL software framework* that supports developers during the implementation of post-WIMP interactive spaces for collaborative knowledge work and enables the realization of ZOIL’s design principles in practice.

3.) The four *example prototypes* based on ZOIL that can serve as exemplars for designers and developers likewise.

Each of the six *ZOIL design principles* is derived from literature of disciplines related to Human-Computer Interaction including Ubiquitous Computing, Information Visualization, Computer-supported Cooperative Work, Cognitive Science, Personal Information Management, and Software Engineering.

The formulation of the *ZOIL design principles* is empirically validated and extended based on the experiences and the findings from own user studies during creating and deploying the four *ZOIL example prototypes* for different application domains (e.g., e-Science, collaborative search, creative design).

The new open-source *ZOIL software framework* serves the implementation of post-WIMP interactive spaces that follow the *ZOIL design principles*. This software framework is described in its architecture and software design patterns and is evaluated in terms of its usability and practical value for developers in an API usability evaluation study.

Acknowledgements

It feels almost impossible to name all the different people who helped to shape this work and directly or indirectly contributed to it. During my years in the Human-Computer Interaction Group at the University of Konstanz, I met and collaborated with so many different inspiring persons in so many different contexts that it is now hard, if not impossible, to remember and name them all or to order them by their importance. Here, I want to thank all of them, including those who I might have forgotten.

First of all, I would like to thank Michael Zöllner and Andreas Engl without whom the *ZOIL prototypes* and the *ZOIL software framework* would not have become reality. I was very fortunate to convince them to work in my project when they were Bachelor students, because they understood my conceptual goals for ZOIL from the beginning and how I wanted these goals to be reflected in software architecture and program code. In their code, they were always willing to go the extra mile of not just making things work but sharing or exceeding my standards for readablility, accessibility, and maintainability. This is particularly true for Michael and his excellent implementation work on ZOIL's data backend, MVVM pattern, and the *Facet-Streams* prototype.

In connection with *Facet-Streams*, I also would like to thank my fellow PhD candidate Jens Gerken. I cannot remember a time as a researcher during which I have been more productive than during the time that Jens, Michael, and I spent at Microsoft Research Cambridge in Winter 2009. *Facet-Streams* would not have become reality without the many days and nights that we have spent discussing, designing, coding, and testing *Facet-Streams* together – and still being able to enjoy Cambridge's pub culture and pub food now and then. Furthermore, throughout my PhD, Jens was always a constant source of invaluable input concerning user studies and experimental designs. Thanks to him, I could study the collaborative use of *Facet-Streams* and ZOIL's API usability under realistic conditions. In a similar fashion, Svenja Leifert was irreplaceable and helped me to greatly refine the initial experimental setup and data analysis of Sören Schubert for my ZUI navigation study. Therefore Svenja and Sören deserve my great gratitude.

There were also many people who helped me to demonstrate ZOIL and its software framework with great prototypes, most notably Mathias Heilig, Mischa Demarmels, Sebastian Rexhausen, Stephan Huber, Oliver Runge, and Benjamin Frantzen. They used the *ZOIL design principles* and an early version of the *ZOIL software framework* to create their own impressive prototypes (e.g., *MedioVis 2.0)* in their Bachelor's, Master's or PhD theses. Furthermore, they helped to create the *Media Seminar Room* prototype. I would also like to thank Jonas Schweizer for his design and implementation of a multi-focus ZUI for ZOIL as part of his Bachelor's thesis that I supervised. Later in the PhD project, it was Florian Geyer and his team of student developers suc as Anita Höchtl or Jochen Budzinski who used the *ZOIL design principles* and pushed the *ZOIL software framework* to its limits and created award-winning ZOIL prototypes such as *Affinity Table*.

Furthermore, I would like to especially thank Toni Schmidt for the implementation of the *DeskPiles* prototype that we deployed at the University of Cambridge and for having the patience for realizing the countless changes that I asked for in course of this project. Toni was also one of the many students from Konstanz who contributed to this thesis by being participants in the API usability evaluation studies of the ZOIL software framework during the two courses "Visual Information Seeking Systems" in the summer terms 2009 and 2010. In particular, I would like to thank the students Till Niese and Marcio Alves de Castro for their impressive *HotelSearch* prototype. I also want to thank Roman Rädle, who had his first contact with ZOIL as a Master's student during one of these courses and later decided to use it in his PhD project. Roman has been my closest collaborator during the final phase of my PhD and I am very thankful and happy that he is actively working on new versions of the ZOIL software framework. I wish him all the best with his PhD and hope that we will continue to work together.

I also want to express my great gratitude to Natasa Milic-Frayling of Microsoft Research who recognized my work's potential and enabled me to further develop my research during my months at Cambridge and also to work with Jeremy Baumberg in the NanoPhotonics Centre of the University of Cambridge. This collaboration and the many inspiring discussions with Natasa and Jeremy resulted in the *Facet-Streams* and *DeskPiles* prototypes, which became key parts of this work and provided me with important publications, many new findings, invaluable feedback, and also revealing experiences of post-WIMP user interfaces "in the wild".

With regard to ZOIL it is also very important to mention two of my fellow students and later colleagues from Konstanz with whom I conceived of many of ZOIL's early ideas when we still were Master's students: Jens Gerken and Werner König. During our time in Konstanz, Jens and Werner were my main source of inspiration and we were endlessly exchanging ideas and feedback about our individual projects and research interests. Although Jens and Werner decided against ZOIL as the focus of their PhD projects, they must be credited for creating the intense atmosphere between enthusiastic creative thought and healthy skepticism in which ZOIL could bloom. I sincerely hope that there will be a chance to work with them again in future.

Last but not least, I would like to thank my advisor Harald Reiterer for the many years of inspiration and guidance and his countless pieces of advice that convinced me to pursue a PhD and an academic career. Since my first steps into research as a Bachelor student until the final phases of my PhD, Harald always motivated me to tackle fundamental research questions beyond the "least publishable unit". With the HCI Group in Konstanz, Harald created a unique environment of independent thinking and work in which my scientific, technical, professional, and personal skills grew with every new challenge. For this reason, I would like to thank Harald for the great opportunities, trust, and the many resources that he provided over the years and I hope that this thesis does justice to them.

To my parents and my siblings
for their unconditional support
and for teaching me about both
the joys and pitfalls of science.

Publications

Some materials, ideas, and figures have appeared previously in the publications listed below.

Journal articles

- **Jetter, Hans-Christian**, Zöllner, Michael, Gerken, Jens, and Reiterer, Harald (2012), 'Design and Implementation of Post-WIMP Distributed User Interfaces with ZOIL', *Intl. Journal of Human-Computer Interaction*, 28 (11), 737-47.
  Text from the article is used in sections 3.4 and 3.5. Figures from the article are used in Figures 2, 12, 16, 19, 24, 26, 27, 44-45, 49, 55.
- Elmqvist, Niklas, Vande Moere, Andrew, **Jetter, Hans-Christian**, Cernea, Daniel, Reiterer, Harald, and Jankun-Kelly, T.J. (2011), 'Fluid Interaction for Information Visualization', *Information Visualization*, 10 (4), 327-40.
  Text from the article is used in sections 2.4 and 6.2.

Conference full papers

- **Jetter, Hans-Christian**, Leifert, Svenja, Gerken, Jens, Schubert, Sören, and Reiterer, Harald (2012), 'Does (Multi-)Touch Aid Users' Spatial Memory and Navigation in 'Panning' and in 'Zooming & Panning' UIs?', Proceedings of the International Working Conference on Advanced Visual Interfaces (AVI '12) (Capri Island, Italy: ACM), 83-90.
  Text from the paper is used in section 4.5. Figures from the paper are used in Figures 72, 96-99, 100-101.
- **Jetter, Hans-Christian**, Gerken, Jens, Zöllner, Michael, Reiterer, Harald, and Milic-Frayling, Natasa (2011), 'Materializing the query with facet-streams: a hybrid surface for collaborative search on tabletops', Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11) (Vancouver BC, Canada: ACM), 3013-22. ***(Honorable Mention Paper Award)***
  Text from the paper is used in section 6.3. Figures from the paper are used in Figures 5, 34-38, 125-130, 133.
- Gerken, Jens, **Jetter, Hans-Christian**, Zöllner, Michael, Mader, Martin, and Reiterer, Harald (2011), 'The concept maps method as a tool to evaluate the usability of APIs', Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11) (Vancouver BC, Canada: ACM), 3373-82.
  A figure from the paper is used in Figure 48.

- **Jetter, Hans-Christian**, Gerken, Jens, Zöllner, Michael, and Reiterer, Harald (2010a), 'Model-based design and implementation of interactive spaces for information interaction', Proceedings of the Third international conference on Human-centred software engineering (HCSE '10) (Reykjavik, Iceland: Springer), 22-37.
  Text from the paper is used in section 3.2. Figures from the paper are used in Figures 12, 16, 19, 24, 42-43, 47, 81-83, 91-92.

Workshop papers

- **Jetter, Hans-Christian**, König, Werner A., Gerken, Jens, and Reiterer, Harald (2008), 'ZOIL - A Cross-Platform User Interface Paradigm for Personal Information Management', The Disappearing Desktop: Personal Information Management 2008 (a CHI 2008 Workshop), *http://nbn-resolving.de/urn:nbn:de:bsz:352-opus-75367*.
  Text from the paper is used in sections 3.1.4. Figures from the paper are used in Figure 45 and Figure 63.

Book sections

- **Jetter, Hans-Christian**, Zöllner, Michael, and Reiterer, Harald (2011), 'Gestaltung und Programmierung von interaktiven Räumen mit dem ZOIL-Paradigma', in Rainer Groh and Martin Zavesky (eds.), Wieder mehr sehen! Aktuelle Einblicke in die Technische Visualistik (Dresden, Germany: TUDpress), 105-15. Figures from the article are used in Figures 16, 19, 21, 26-27, 29, 45, 91-92, 95.

- Zöllner, Michael, **Jetter, Hans-Christian**, and Reiterer, Harald (2011), 'ZOIL: A Design Paradigm and Software Framework for Post-WIMP Distributed User Interfaces', in José A. Gallud, Ricardo Tesoriero, and Victor M. R. Penichet (eds.), Distributed User Interfaces: Designing Interfaces for the Distributed Ecosystem (Human-Computer Interaction Series: Springer London), 87-94.
  Text from the article is used in sections 3.4.2 to 3.4.6. Figures from the article are used in Figure 45 and Figure 47.

- Seifried, Thomas, **Jetter, Hans-Christian**, Haller, Michael, and Reiterer, Harald (2011), 'Lessons Learned from the Design and Implementation of Distributed Post-WIMP User Interfaces', in J. A. Gallud, R. Tesoriero, and V. M. R. Penichet (eds.), Distributed User Interfaces: Designing Interfaces for the Distributed Ecosystem (Human-Computer Interaction Series: Springer London), 95-102. A figure from the article is used in Figure 5.

Posters

- Geyer, Florian, **Jetter, Hans-Christian**, Pfeil, Ulrike, and Reiterer, Harald (2010), 'Collaborative Sketching with Distributed Displays and Multimodal Interfaces', Proceedings of ACM International Conference on Interactive Tabletops and Surfaces (ITS ‘10) (Poster Session) (Saarbrücken, Germany: ACM), 259-60.
  A figure from the paper is used in Figure 30.

Contributing theses

Following Bachelor’s theses that I supervised contributed to the technical implementation of the ZOIL software framework (see section 3.4 for details):

- Zöllner, Michael (2009), 'Ein persistentes objektorientiertes Datenmodell für das Personal Information Management in ZOIL', Bachelor's Thesis (University of Konstanz).
  No text or figures from the Bachelor’s thesis are used in this dissertation.
- Engl, Andreas (2008), 'A Framework for an Infinitely Zoomable Information Landscape', Bachelor's Thesis (University of Konstanz).
  A figure from the Bachelor’s thesis is used in Figure 95.

Following Bachelor’s thesis that I supervised demonstrated how to implement portals and multi-focus ZUIs with the ZOIL software framework (see section 4.3.6 for details):

- Schweizer, Jonas (2009), 'Gestaltung und Implementierung eines Multi-Fokus und Multi-Display Management-Systems', Bachelor's Thesis (University of Konstanz).
  Figures from the Bachelor’s thesis are used in Figures 85-87.

Following Diploma thesis and Master’s thesis that I supervised contributed to the user study of touch vs. mouse navigation in ZUIs (see section 4.5 for details):

- Leifert, Svenja (2013), 'Theorie und experimentelle Untersuchung des Einflusses von Interaktionsdesign auf das räumliche und inhaltliche Gedächtnis', Diploma Thesis (University of Konstanz).
  No text or figures from the Diploma thesis are used in this dissertation.
- Schubert, Sören (2010), 'Auswirkung verschiedener Eingabegeräte auf das Erlernen räumlich-visueller Merkmale einer virtuellen Informationslandschaft', Master's Thesis (University of Konstanz).
  No text or figures from the Master’s thesis are used in this dissertation.

List of Abbreviations

2D	two-dimensional
3D	three-dimensional
ACID	Atomicity, Consistency, Isolation, Durability
API	Application Programming Interface
CASE	Computer-aided Software Engineering
CPU	Central Processing Unit
GUI	Graphical User Interface
GPU	Graphics Processing Unit
HCI	Human-Computer Interaction
IDE	Integrated Development Environment
IV, InfoVis	Information Visualization
MIT	Massachusetts Institute of Technology
OOUI	Object-Oriented User Interface
PARC	(Xerox) Palo Alto Research Center
PDA	Personal Digital Assistant
post-WIMP	post-“Windows, Icons, Menu, Pointer”
SD	Standard Deviation
SDK	Software Development Kit
SDMS	Spatial Data Management System
SQL	Structured Query Language
TCP/IP	Transmission Control Protocol/Internet Protocol
TUI	Tangible User Interface
UI	User Interface
VR	Virtual Reality
WIMP	“Windows, Icons, Menu, Pointer”
WWW	World Wide Web
XAML	Extensible Application Markup Language
ZOIL	Zoomable Object-Oriented Information Landscape

Contents

List of Figures

"I believe the successor to the desktop is the room,
that instead of thinking that the computer is just
something on the desk that you go and sit in front of,
in the future basically the whole room is the computer
and you go in it."

Craig Mundie, Microsoft Chief Research
and Strategy Officer, 2011

1 Introduction

Imagine a keynote address in the year 2070 celebrating the 100th anniversary of the microprocessor. The topic of this keynote is the history of computing and the enormous impact computing technology had on all societies around the world – if not on mankind itself. What do you believe will the speaker consider as the most important idea in the history of computing?

Without any doubt, computers would not exist without the mathematical, algorithmic, and architectural foundations laid between the early 19th century until the mid of the 20th century by mathematicians and engineers such as Charles Babbage, Ada Lovelace, Alan Turing, Konrad Zuse, or John von Neumann (*Goldstine 1972*). However, their achievements had the strongest impact only after computers became *"personal dynamic media"* for communication and creative thought, a part of the Dynabook vision of Alan Kay and his colleagues from Xerox PARC of the 1970s and 1980s that initiated the era of personal desktop and laptop computers (*Kay and Goldberg 1977*). Others might argue that the golden age of computing truly began only in the 1990s with Tim Berners-Lee's 'World Wide Web' that made vast amounts of information and services accessible as Hypertext on billions of networked devices. Yet, to achieve this, Berners-Lee needed a worldwide communication network called the 'Internet' that was shaped by Vint Cerf, Bob Kahn, and Leonard Kleinrock in the 1970s (*Leiner, Cerf, Clark et al. 1997*). And how usable would this World Wide Web have been without user interface pioneers of the 1960s such as Ivan Sutherland or Douglas Engelbart (*Brad A Myers 1998*), who invented the graphical user interface and mouse interaction with Hypertext and thus paved the way for a adoption of computing technology among a broad user population and not only among scientists and engineers?

Of course I do not claim that I know the "correct" answer to the question about the most important idea of computing history; neither do I claim that there is one. However, I believe that this thought experiment reveals an important pattern in computing history. Beginning with the 1950s, most computing pioneers began to share a vision that became tremendously influential and resulted in deep societal changes: If computers – once room-sized technological curiosities helping to decrypt the enemy's secret communications or to calculate rocket trajectories – get powerful, connected, usable, and

small enough, they can become a ubiquitous personal tool on the desks and in the pockets of every person on the planet. By giving computers an accessible and usable form, they can become a creative *"metamedium"* that enables animators, musicians, high school students, or even young children to edit or create documents, drawings, paintings, animation, and music (*Kay and Goldberg 1977*). Computers can become *"tools for thought"* and, according to J. C. R. Licklider, a universal medium that, like literacy, could become the property of the entire culture and lead to a boost in human cultural capabilities (*Rheingold 2000*).

Driven by the societal and economic prospects of a personal, mobile, and networked computing for everyone, researchers and engineers achieved an exponential growth in performance and network bandwidth while minimizing space and energy consumption and increasing computer usability. One of the most striking results of this technological progress is that the number of computerized mobile-connected devices will exceed the number of people on earth by the end of 2012[1].

1.1 The Vision of Ubiquitous Computing

In 1991, Mark Weiser and his colleagues from Xerox PARC identified and interpolated this trajectory of computing history to envision the computer for the 21st century. Instead of a single multi-purpose computer, they envisioned a world of *"ubiquitous computing"* (*Weiser 1991*) with many new breeds of portable and stationary devices. For example, Weiser's "tabs", "pads", and "boards" fit seamlessly into our existing work practices because their form factors and user interfaces are inspired by familiar tools, e.g., by writing on whiteboards or notepads. Weiser argued that *"real power (...) comes not from any of these devices – it emerges from the interaction of all of them"* (*Weiser 1991*). This illustrates a global technological, societal, and economic change: The virtual world of bits and bytes leaves its traditional habitat of research labs and office desks and becomes an integrated part of our physical and social environment to serve our information needs.

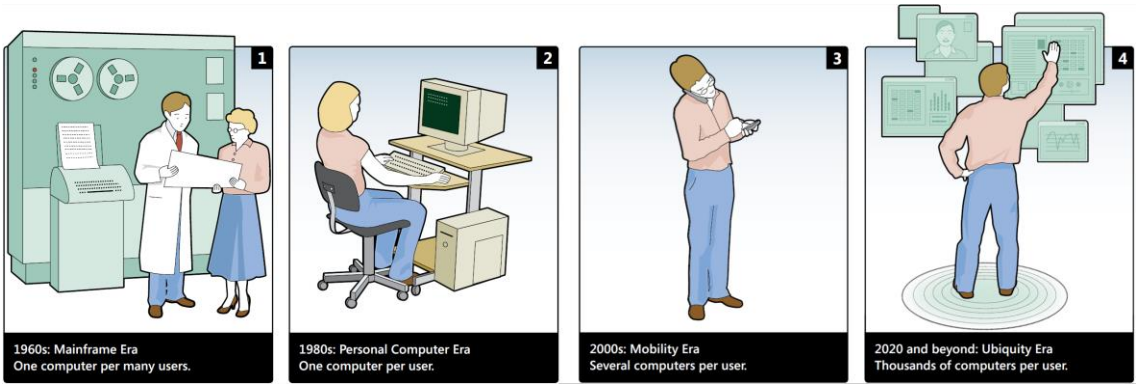


Figure 1 – The history of computing from the mainframe era of the past to the ubiquity era of the future. Source: (*Harper, Rodden, Rogers et al. 2008*).

[1] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2011-2016 *http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.html* (Accessed Jun 20, 2012).

Thereby Weiser's goal for ubiquitous computing was to entirely rethink technology to really serve us and not the opposite. *"Machines that fit the human environment instead of forcing humans to enter theirs will make using a computer as refreshing as taking a walk in the woods"* (*Weiser 1991*). In Weiser's vision, computing becomes an invisible tool in our natural environment and using it involves less strain and fewer *"mental gymnastics"*, so that we are freed to use it without thinking and to focus beyond it on new goals. It helps us to overcome the problem of information overload and poses no barrier to personal interactions, but brings communities together.

1.2 The Reality of Ubiquitous Computing

While the fascinating prospects of ubiquitous computing have motivated generations of computer scientists, engineers, and designers to advance the field, the present-day practice of ubiquitous computing (ubicomp) is still disillusioning, hardly usable for novices, and needs fundamental changes. Even over 20 years after its formulation, the original Weiserian vision of computers that vanish into the background has not turned into reality. Nevertheless, computers are already deeply woven into the fabric of our physical and social environment. Often they define the way we communicate and work instead that they enable us to communicate and work in the way we want.

For example, Bell and Dourish argue that ubicomp is already here, although not in the form ubicomp researchers like Weiser originally envisaged (*Bell and Dourish 2006*): Interacting with the ubicomp of the present is far less seamless and more heterogeneous than in the Weiserian vision. While researchers and technologists continue to conjure a vision of ubicomp for the proximate future, they treat present-day problems *"as implementation issues that are, essentially, someone else's problem, to be cleaned up afterwards as part of the broad march of technology"*. Bell and Dourish suggest that dealing with the *"messiness"* of present-day ubicomp should become a central element of ubicomp research instead of hoping for future standardization and consistency.

Oulasvirta observed users 'do' the ubicomp. He characterizes present-day ubicomp as *"a multilayered agglomeration of connections and data, distributed physically and digitally, and operating under no recognizable guiding principles"* (*Oulasvirta 2008*). He regards *"achieving seamlessness"* and *"fluent multidevice work"* as key challenges. *"The drifting apart of HCI research and real-world ubicomp is worrisome because improving the state of affairs is not the duty of engineers alone."* In 2007, Huuskonen gave similar reasons for predicting a *"ubiquitous computing meltdown"* in the next decade unless major shortcomings such as usability and interoperability of integrated systems are solved (*Huuskonen 2007*). More recently, Greenberg et al. described interconnecting, configuring, and debugging present-day digital ecologies of interactive devices as painful and time consuming (*Greenberg, Marquardt, and Ballendat 2011*): Today's devices are still far from seamless and performing tasks among then is tedious, for example navigating through network and local folders to find and exchange files.

The motivation for this thesis arises from this unsatisfying reality of present-day ubicomp. My goal for this thesis is to contribute a small piece to the overall improvement of present-day ubicomp by addressing a specific problem set:

> *How can designers and developers of ubiquitous computing environments be supported to create more usable multi-user and multi-device post-WIMP interactive spaces for co-located collaborative knowledge work?*

In the remainder of this chapter, I provide a more detailed description of this problem set and its context by introducing the necessary definitions. After this, I formulate my research goal and the scope of this thesis in detail.

1.3 Definition: Post-WIMP Interactive Spaces

In his seminal publication, Mark Weiser envisioned a room in which multiple users gather around a large pen-operated *"live board"* that serves as a digital equivalent to chalkboards or whiteboards (*Weiser 1991*). This board is integrated with other digital devices in the room, e.g., small active badges worn by the users, pen-operated portable tablets, or desktop devices for accessing and editing digital content (Figure 2, left).

This vision of augmenting a physical room with computer technology to enable co-located groups of users to collaborate became a driving force in the field of ubiquitous computing and HCI research. For example, the vision of the i-LAND environment of Streitz et al. with *"roomware"* devices such as an interactive electronic wall with chairs ('DynaWall' and 'CommChairs', Figure 2, right) (*Streitz, Geißler, Holmer et al. 1999*; *Streitz, Tandler, Müller-Tomfelde et al. 2001*) or an interactive table ('ConnecTable') (*Tandler, Prante, Müller-Tomfelde et al. 2001*) has inspired the work of many ubicomp researchers.

Figure 2 – The vision of a computer-augmented room of Mark Weiser (left). Source: (*Weiser 1991*). DynaWall and CommChairs in i-LAND (right). Source: (*Streitz, Tandler, Müller-Tomfelde et al. 2001*).

A further famous example is the iRoom project at Stanford University that combined a four-projector tiled display with an interactive table to investigate human interaction

with large high-resolution displays (Figure 3, left) and then began to design and use rooms (Figure 3, right) containing one or more large displays that had the ability to integrate portable devices (*Johanson, Fox, and Winograd 2002*). The iRoom is the source of a broad range of ubicomp interaction techniques, software architectures, and one of the rare examples of a ubicomp system that was in everyday use over several years[2].

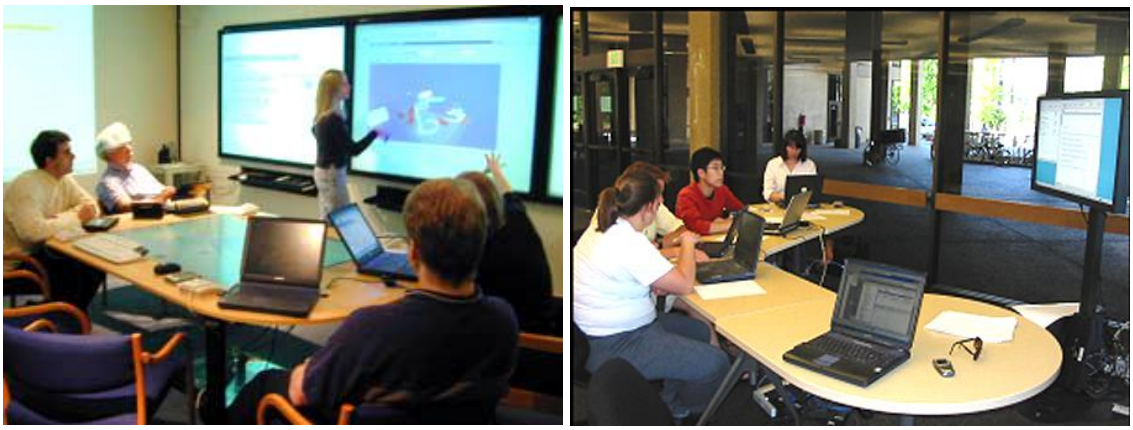


Figure 3 – Examples of different usage scenarios of Stanfords iRoom.
Sources: (*Johanson, Hutchins, Winograd et al. 2002*; *Shih, Crone, Fox et al. 2004*).

Figure 4 – The 'Future Meeting Room' or ICE (Interactive Collaborative Environment).
Source: (*Benyon and Mival 2012*).

Since iRoom and i-LAND, new technologies and devices have resulted in an ongoing interest of researchers in computer-augmented physical rooms as collaborative environments. For example, the advent of interactive multi-user tabletops with multi-touch input (*Dietz and Leigh 2001*) resulted in research on table-centric environments for ubiquitous computing (*Benyon and Mival 2012*; *Wigdor, Shen, Forlines et al. 2006*). Today, the availability of commercial tabletop products that are able to track physical objects as tangible user interface elements (e.g., Microsoft Surface since 2008, Samsung SUR40 with

[2] *http://net.educause.edu/ir/library/pdf/P7102cs22.pdf* (Accessed Jun 25, 2012).

Microsoft PixelSense[3] since 2011) and their interplay with powerful mobile devices continues to create a steady stream of publications and prototypes. For example, as a part of this thesis, I have been working on tabletop user interfaces that integrate tablet PCs (Figure 5, left) or employ tangible user interface elements for faceted collaborative search (Figure 5, right) (*Jetter, Gerken, Zöllner et al. 2011*).

Figure 5 – The NiCE Discussion Room et al. (top). Multi-user and multi-device collaboration in e-Science (left). *Facet-Streams* for collaborative search (right). Source: (*Seifried, Jetter, Haller et al. 2011*).

Anoto's digital pen & paper technology[4] now enables high-precision multi-user pen input, for example on large displays in meeting rooms for creative work (*Haller, Leitner, Seifried et al. 2010*) (Figure 5, top). This raises many practical questions for interaction design, e.g., how to design undo/redo functionality for a system, where users work in one workspace at different locations at the same time (*Seifried, Rendl, Haller et al. 2012*). This makes it necessary to make systems aware of the current positions of all users, for

[3] *http://www.microsoft.com/en-us/pixelsense/default.aspx* (Accessed Jul 20, 2012).
[4] *http://www.anoto.com/* (Accessed Jun 25, 2012).

example by using novel low-cost tracking technologies like the Microsoft Kinect[5]. This is also important for *proxemic interactions (Greenberg, Marquardt, and Ballendat 2011), a* recent approach of interaction design for ubicomp digital ecologies that is based on devices that are aware of nearby people and devices and their position, identity, movement, and orientation.

1.3.1 Interactive Spaces

The above examples have illustrated the typical characteristics of the ubiquitous computing environments that I focus on in this thesis:

- *Multiple users* and *multiple devices* are *co-located* in a single *physical space* or *room*. The physical environment and the contained computing technology are designed for the purpose of *collaboration* between the multiple users who are currently present in the physical space.
- The devices in the environment can either be *stationary*, for example large vertical displays or horizontal interactive tabletop computers that are permanently situated in the room, or *mobile*, for example laptop computers, tablet PCs, or smart phones, that are permanently or only temporarily available in the room.
- The devices can be *input devices* without own output, e.g., digital pens or mice, *output devices* without input, e.g., non-touch-enabled displays or projectors, or *input/output devices* with the ability to process user input and react to it with system output, e.g., interactive tabletops or touch-enabled displays that are connected to a personal computer.
- *Active devices* have underlying computational resources to execute application software. For example, a tabletop like the Microsoft Surface contains personal computing hardware with a CPU, main memory, and a graphics board and can execute applications for the Windows operating system. On the contrary, *passive devices* execute only closed embedded software. For example, mice and digital pens need embedded software to process sensor input, or displays and projectors need embedded firmware to provide visual output and signal processing.

Within this thesis, I refer to a ubiquitous computing environment with the above characteristics as an *interactive space.* The use of this term was inspired by Streitz et al.'s *"interactive landscape"* (*Streitz, Geißler, Holmer et al. 1999*), Biehl et al.'s *"application relocation in an interactive space"* (*Biehl and Bailey 2004*), and Wigdor et al.'s *"table-centric interactive spaces for real-time collaboration"* (*Wigdor, Shen, Forlines et al. 2006*). It is the result of a conscious decision against terms such as *smart room, intelligent room,* or *smart environment,* that are frequently used in the field of Ambient Intelligence (*Aarts 2003*; *Cook and Das 2004*). This decision is based on the expectations that the words

[5] *http://www.microsoft.com/en-us/kinectforwindows/* (Accessed Jun 25, 2012).

"smart" or "intelligent" raise. Unlike the vision of smart or intelligent technology that uses action prediction and identification to anticipate user needs and thus puts a focus on reacting to users' *implicit interaction,* e.g., users' natural movement in a room, this thesis focuses on the design and implementation of technology that processes and reacts to the *explicit interactions* of users (*Ju and Leifer 2008*), e.g., direct manipulation of objects on a tabletop. While it appears as a very promising and worthwhile effort to combine smart or intelligent approaches for *implicit* or *proxemic interactions* (*Greenberg, Marquardt, and Ballendat 2011*) with the designs and technologies introduced in this thesis, they were not part of my research. However, such combined approaches are currently investigated by Roman Rädle in our HCI group at the University of Konstanz.

1.3.2 Post-WIMP Interaction

WIMP is an acronym that describes the key components of the dominant form of human-computer interaction of the past decades: ***W****indows,* ***I****cons,* ***M****enus,* ***P****ointer*. Since the advent of the Graphical User Interface (GUI) in the late 1970s, WIMP interaction with the desktop metaphor has become the dominant technological paradigm for user interfaces. It has been remarkably successful in making computing technology accessible to a broad user population and its great commercial success has thereby set a de-facto standard for user interfaces in personal computing, also influencing other strands of devices, e.g., multi-purpose mobile devices such as Pocket PCs or PDAs.

However, since the arrival of the World Wide Web, the desktop metaphor and its simulation of an office environment using metaphorical files and folder hierarchies that can be navigated, viewed, and edited inside of application windows are increasingly criticized for being inappropriate for today's contexts of use and the eras of mobility and ubiquity (*Moran and Zhai 2007*; *Müller-Prove and Ludolph 2007*; *Ravasio and Tscherter 2007*; *Voida, Mynatt, and Edwards 2008*). *"Today's greater set of physical interaction devices and modalities"*, the *"multiplicity of devices"*, and the importance of *"social interaction"* make it necessary to rethink the dominant designs of our user interfaces such as the desktop metaphor (*Moran and Zhai 2007*).

In 1997, van Dam coined the term *post-WIMP user interfaces* to refer to the numerous emerging interaction styles that provide alternatives to the established ways how user interfaces present information and how we can interact with it: *"A post-WIMP interface to me is one containing at least one interaction technique not dependent on classical 2D widgets such as menus and icons. Ultimately it will involve all senses in parallel, natural language communication and multiple users"* (*van Dam 1997*). As examples for post-WIMP user interfaces, van Dam mentions pen-based input on PDAs, alternative input devices such as steering wheels or golf clubs for arcade games, marking menus, or two-handed input using the dominant and non-dominant hand simultaneously.

Ten years later, Jacob et al. proposed the notion of *Reality-based Interaction* as a unifying concept to tie together emerging post-WIMP interaction styles and to provide a framework that can be used to understand, compare, and relate the current paths of HCI research on post-WIMP interaction (*Jacob, Girouard, Hirshfield et al. 2007*; *Jacob,*

Girouard, Hirshfield et al. 2008). After analyzing the presentation and interaction styles of many different post-WIMP user interfaces (e.g., the Apple iPhone with spatial "cover flows" and multi-touch and accelerometer input, tangible and multi-touch interaction on tabletops, stereoscopic VR systems with gestural interaction for picking up or dropping virtual objects), they suggested four *"themes of reality"* on which successful post-WIMP user interfaces build and which have been neglected in WIMP UIs so far (Figure 6). By employing these four themes of reality and building on users' pre-existing knowledge of the everyday, non-digital world to a much greater extent than before, post-WIMP interfaces attempt to make computer interaction more like interacting with the real, non-digital world (*Jacob, Girouard, Hirshfield et al. 2008*).

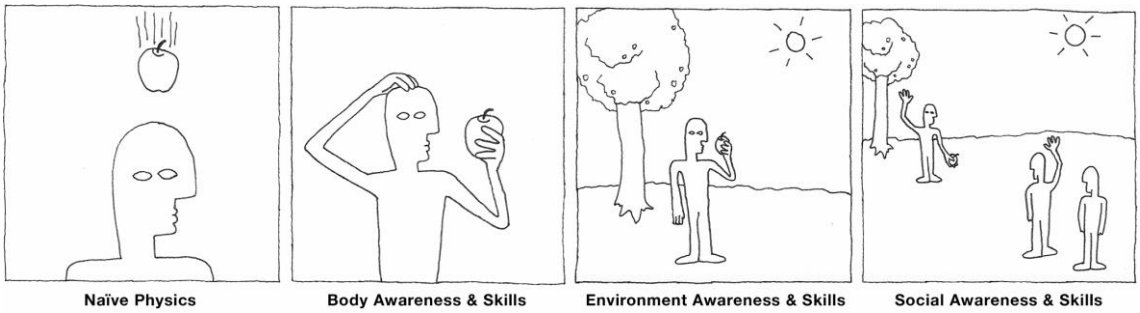


Figure 6 – Jacob et al.'s four *themes of reality* for understanding and designing post-WIMP interaction. Source: (*Jacob, Girouard, Hirshfield et al. 2008*).

1.) *Naïve Physics (NP)*: Users' informal human perception of basic physical principles or users' common sense knowledge about the physical world (e.g., gravity, friction, velocity, the persistence of objects).

2.) *Body Awareness and Skills (BAS):* Users' awareness of their own physical bodies and users' skills for controlling and coordinating their bodies (e.g., proprioception, coordinated movements of limbs, head, eyes).

3.) *Environment Awareness and Skills (EAS)*: Users' awareness of their physical surrounding and users' skills for navigating and manipulating this environment.

4.) *Social Awareness and Skills (SAS):* Users' awareness of the presence of others and users' skills for social interaction (e.g., non-verbal communication, ability to exchange physical objects and to collaborate on a task).

When looking at the interactive spaces introduced in section 1.3, it is difficult to clearly distinguish between WIMP and post-WIMP interaction. They combine techniques of post-WIMP interaction such as pen-based input (an example for Jacob et al.'s *BAS* theme) and multi-user collaboration (an example for Jacob et al.'s *SAS* theme) with traditional WIMP concepts such as applications, window management, files, and folders. This is not surprising given the dominance of WIMP in today's digital ecologies. It is difficult to integrate an interactive space into existing workflows without using traditional WIMP concepts such as application software, application-specific file formats, and folder hierarchies. Often the use of established WIMP applications such as word processors,

presentation software, or graphic editors inside the interactive space is necessary. Therefore most research in the field has focused on expanding or extending current WIMP interface paradigms (e.g., applications, windows, menus, pointers) which might not necessarily be optimal for multi-display environments or interactive spaces (*Nacenta 2008*). In contrast, Nacenta suggests working on a new breed of interfaces that can revolutionize collaboration in co-located spaces and depart significantly from current established interaction techniques and standard WIMP interfaces.

This thesis follows such an approach and tries to support actual use scenarios while entirely rethinking interaction based on the critical voices in HCI literature that have called for a replacement of WIMP and the desktop metaphor with newer post-WIMP approaches. As Nacenta discusses, such an "revolutionary" approach involves great technical challenges since researchers have to create prototypes of these new paradigms *"that do not take a lifetime to build, but provide enough fidelity that will allow us to evaluate the merit of the idea"* (*Nacenta 2008*). For this reason, this thesis focuses on prototyping and evaluating only selected post-WIMP interaction techniques. The notion of post-WIMP interaction in this thesis can therefore be summarized as follows:

- To achieve a seamless and natural interaction, this thesis strives for a closely integrated work environment without application boundaries that is based on *object-oriented* instead of *application-oriented user interfaces* (see chapter 3).
- Information items are stored and accessed in a shared spatial workspace that serves as a unifying visual meta-layer for information resources from the file system or the Web. Ideally, there is no visible use of application windows or browsers and no need for window management and navigating folder hierarchies. Instead, all kinds of information items and tools are organized and manipulated inside of a *zoomable user interface* (ZUI) and can be directly accessed using *semantic zooming* (see chapter 4).
- This thesis explores different approaches for the use of the *virtual space* in a ZUI to support sensemaking, marks, and annotations. Furthermore, *physical space* on a tabletop and *virtual space* in a ZUI are used to support different collaboration styles. In particular, the different active input/output devices in the interactive space can serve as individual cameras into the shared zoomable workspace to support mixed-focus collaboration[6] (see chapter 5).
- This thesis explores the use of post-WIMP *information visualization* (or *InfoVis*) for a *fluid interaction* with and efficient access to analytical overviews of the information space. Users are enabled to collaboratively search and browse in collections of items using *faceted navigation* and virtual or tangible lenses with visualizations of meta-data, e.g., lists, maps, scatter plots (see chapters 4 & 6).

[6] More details and definitions for this nature of collaboration are formulated in the next section.

- Horizontal interactive tabletop computers promise many advantages for co-located collaboration. This thesis explores a *hybrid visual-tangible externalization* of a filter/flow metaphor on a tabletop with multi-touch and tangible input. It enables *fluid interaction* with post-WIMP InfoVis for collaborative "around-the-table" faceted search (see chapter 6).

1.4 Definition: Collaboration

As mentioned, the purpose of the post-WIMP interactive spaces in this thesis is *collaboration* among multiple users and thus these systems can be considered as an example of *groupware* (*Ellis, Gibbs, and Rein 1991*). To describe the nature of this groupware and the intended collaboration therein, this section uses different models from the field of Computer-supported Cooperative Work (CSCW) which *"can (...) be considered as a scientific discipline guiding the design and development of groupware in a meticulous and appropriate way"* (*Greenberg 1989*).

1.4.1 3C Collaboration Model

The 3C collaboration model of Fuks et al. is based on pioneering work of Ellis et al. (*Ellis, Gibbs, and Rein 1991*) and considers computational support for collaboration as the interplay between *communication*, *coordination*, and *cooperation* tools. *"Communication is related to the exchange of messages and information among people, coordination is related to the management of people, their activities and resources; and cooperation is the production taking place on a shared workspace"* (*Fuks, Raposo, Gerosa et al. 2008*).

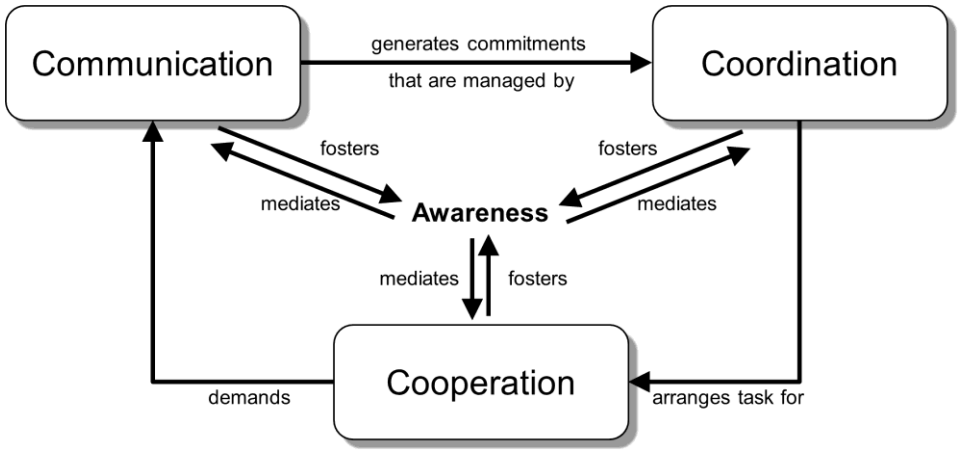


Figure 7 – The 3C Collaboration Model. Figure adapted from (*Fuks, Raposo, Gerosa et al. 2008*).

In the context of this thesis, ***cooperation*** is the joint operation of members of the group inside an interactive space, seeking to execute tasks, and generate and manipulate cooperation objects. Thereby the tasks are different activities of collaborative *knowledge work*. I provide a more detailed description of knowledge work for the context of this thesis in section 1.5. For now, it is sufficient to consider it as different activities of knowledge-intensive work, e.g., search, sensemaking, and the creation of new information artifacts.

Cooperation demands ***communication***. The communication between users of the interactive spaces in this thesis is based on the natural social protocols and the informal communication of the everyday non-digital world and is not mediated by technology. Since users are co-located in a physical environment they can use the full range of verbal and non-verbal communication, including gestures, pointing, and similar natural styles of communication. However, the efficiency of this communication can be largely improved by providing appropriate technology-supported externalizations, e.g., virtual notes or annotations, shared visual maps, or tangible user interface elements on a tabletop. Externalizations of this kind use physical space and proximity to express the current state of work and give the group a shared overview. They also provide a shared frame of reference for efficient communication, e.g., by pointing, and help to avoid imposing rigid work or communication patterns on users. For example, chapter 6 discusses how a visual and tangible externalization of a search process can lead to a more efficient communication.

By improving communication and giving an overview of the state of collaboration, externalizations also help to create the necessary ***awareness*** within the group. Awareness *"is an understanding of the activities of others, which provides a context for your own activity. This context is used to ensure that individual contributions are relevant to the group's activity as a whole, and to evaluate individual actions with respect to group goals and progress. The information, then, allows groups to manage the process of collaborative working"* (*Dourish and Bellotti 1992*).

This management of the process of collaborative working is called ***coordination***. Coordination is the link between cooperation and communication in order to enforce the success of collaboration (*Fuks, Raposo, Gerosa et al. 2008*). It includes coordination of people, tasks, and how these tasks are executed. As discussed in section 1.4.3 and chapters 5 and 6, it is an important design goal for groupware that the coordination can happen flexibly and that seamless switches between different working and coupling styles can be realized by the users.

1.4.2 Time-Space Matrix

A popular model for classifying groupware systems is the time-space matrix introduced by Robert Johansen in (*Johansen 1988*). The matrix considers collaboration along two dimensions: *time* and *space*. As highlighted in section 1.3.1 (p.7), the collaboration in the interactive spaces of this thesis is *co-located* and *synchronous*. This means, that the users are co-located in space, i.e., the physical environment or room of the interactive space, and in time, i.e., they work together at the same time.

The employed technology in this thesis is based on a shared object space using client-server architectures (see section 3.4.1, p.71) and thus can in principle also enable scenarios of remote collaboration. However, the many design and technological challenges that spatially distributed interactive spaces pose, e.g., maintaining awareness, natural communication, and social protocols across remote locations, have not been part of this work.

	Same time (synchronous)	**Different time** (asynchronous)
Same place (co-located)	shared table, single display groupware, ***interactive spaces***	project management, shift work groupware, large public display
Different place (remote)	video conferencing, chats, virtual worlds, shared remote desktop	email, bulletin boards, wikis

Figure 8 – The time-space matrix with different kinds of groupware. Adapted from (*Johansen 1988*) and Wikipedia[7].

1.4.3 Tightly-coupled Collaboration vs. Loosely-coupled Parallel Work

Many collaborative activities, such as brainstorming, designing, and planning, involve *mixed-focus collaboration*, where users frequently transition between individual and shared tasks within a group (*Tang, Tory, Po et al. 2006*). *"A group's collaborative coupling style (henceforth coupling), or the manner in which collaborators are involved and occupied with each other's work, frequently changes (…). For instance, an individual might work on an idea alone before presenting it to the group, and then later work with the group to jointly manipulate the idea"* (*Tang, Tory, Po et al. 2006*). Morris et al. refer to these two styles of mixed-focus collaboration as *tightly-coupled collaborations* vs. *loosely-coupled parallel work* (*Morris, Fisher, and Wigdor 2010*).

In this thesis, collaborative knowledge work is *mixed-focus collaboration* and therefore the interactive spaces need to support tightly-coupled collaborations and loosely-coupled parallel work. Achieving this support is particularly challenging, because the groupware must support both individual and group needs, which are often in opposition (*Gutwin and Greenberg 1999*). For instance, Tang et al. raise the question whether individuals should be able to control how parts of the workspace are viewed, or whether the group should be restricted to a singular view. While independent views may support individual tasks, they may also negatively affect a group's ability to coordinate its activities and manage shared resources (*Tang, Tory, Po et al. 2006*). Furthermore, *fluid transitions* between both working styles during collaboration should be supported by the interaction design of the system, since groups frequently and fluidly transition between several stages of working closely together and working independently.

As a consequence, Tang et al. formulate four implications for design that chapters 5 and 6 employ as requirements for the design of interactive spaces (*Tang, Tory, Po et al. 2006*):

1. Support a flexible variety of coupling styles.
2. Provide fluid transitions between coupling styles.
3. Provide mobile high resolution personal territories.
4. Support lightweight annotations.

[7] *http://en.wikipedia.org/wiki/CSCW* (Accessed Jun 26, 2012).

1.5 Definition: Knowledge Work

The previous sections have outlined the focus of this thesis as post-WIMP multi-user and multi-device interactive spaces that are designed for the purpose of co-located synchronous collaboration. This collaboration is mixed-focus collaboration that includes phases of tightly-coupled collaboration vs. loosely-coupled parallel work. However, the goal of the collaborative activity is still undefined. This section describes this goal in terms of typical activities and tasks. To achieve this, it introduces *knowledge work* as a generic concept that describes the commonalities between the different collaborative activities that are treated in this thesis.

The starting point for the concept of *knowledge work* is the term *"knowledge worker"* popularized by Drucker in 1973 (*Drucker 1973*). It describes the role of a growing percentage of employees in business organizations who put to work what they have learned in systematic education, i.e., concepts, ideas, and theories, rather than employees, who put to work manual skill or muscle (*Kidd 1994*). Since information technology has tended to *"automate away"* the routine, repetitive, and non-adaptive processes of *"production work"* or *"pushing paper"* (*Collins 1995: 29*), it left the unstructured components (e.g., decision-making) to this new kind of worker. Thus knowledge workers need new tools to simulate, visualize, and evaluate alternatives based on data and assumptions (*Collins 1995: 30*).

To characterize the distinguishing behavior of such knowledge workers clearly enough to design appropriate computer tools for them, Kidd interviewed twelve knowledge workers from areas such as design, advertising, marketing, management consultancy, broadcasting, law, finance, and research (*Kidd 1994*). On this basis, she formulated implications for the design of systems supporting knowledge work. The most relevant implications for this thesis can be summarized as follows:

1. Companies value knowledge workers for their diversity. Knowledge workers solve problems and generate outputs largely by resort to structures internal to themselves. Each knowledge worker develops a different internal "configuration" based on changes in their thinking and outlook by the situations they have encountered, the information they have absorbed, and the particular way they have made sense of these. Software for knowledge workers should be careful to provide tools which enable diversification instead of leveling or standardizing individual differences.
2. Software for knowledge workers should avoid trying to "understand" the information it is holding or trying to predict what the user wants to do with it. True knowledge work cannot be automated. At the points where it apparently *can* be automated, then it is no longer true knowledge work.
3. Software for knowledge workers should capture and reproduce the appearance of marks made by knowledge workers on paper, screens, walls, whiteboards, or the physical environment in general. Spatial layout and materials are important

> for knowledge work. Computers for knowledge work should mimic and extend the ability of the physical environment to inform an individual worker or an organization of such workers.

Collins shares Kidd's views on the need for diversification and the dangers of automation for knowledge work (*Collins 1995: 30-31*). He discusses how more and more of the job content of knowledge workers is decision-making. Since decision support tasks are unstructured, the user interface cannot provide the structure that is lacking in the task itself. *"The interface can, however, provide a context for interaction to guide the user through a space of possible actions and results. The context is a rich, graphical representation of the data or other objects of interest, and a means of showing users what actions are possible at each point in the process"* (*Collins 1995: 31*).

While Kidd and Collins tell something about *how* the tools for knowledge work should look like, they do not reveal *what* typical activities they should support. This question can be answered by looking into different theoretical models and frameworks of creative and knowledge-intensive work that exist in literature on human-computer interaction, information visualization, and personal information management such as (*Blandford and Attfield 2010*; *Card 2008*; *Lehikoinen, Aaltonen, Huuskonen et al. 2007*; *Shneiderman 2002*). Although most of these models and frameworks were not explicitly formulated for "knowledge work", they can be considered as typical examples for the kind of knowledge work that happens in our professional and private lives. They therefore outline and describe the scope and nature of the different activities, tasks, and goals that ideally should be supported in the interactive spaces of this thesis.

1.5.1 Blandford and Attfield's "Information Journey"

The *"Information Journey"* is a framework for reasoning about information interaction based on studies of what people really do and how information integrates with their professional and personal lives. The aim has not been simply to understand information work but to develop theory that can inform design and deployment of future technologies (*Blandford and Attfield 2010: 30*). It encapsulates phases of:

- *Recognizing an information need* (also called an "anomalous state of knowledge")

  Examples: 1.) A patient needs health information about a recognized symptom. 2.) A journalist needs to have information to support a particular interpretation or 'angle' taken on a recent event.

- *Acquiring information* (possibly through active searching, or maybe by serendipitous finding or being told)

  Examples: 1.) A patient searches or browses in online health information. 2.) A journalist searches or browses in specialist news archives and general web resources.

- *Interpreting, and often validating, that information*

Examples: 1.) A patient validates if the information is actually advertising, promoting products, or whether it is the opinions of another lay person. 2.) A journalist triangulates information from multiple sources, e.g., competing journalists.

- *Using the interpretation* (e.g., in writing or decision making)

  Examples: 1.) A patient makes a decision about whether to consult a doctor. 2.) A journalist writes an article.

According to Blandford and Attfield, *"these phases are not necessarily sequential; for example, information may be acquired incidentally (without the individual having previously recognized the need), and it may be necessary to find and interpret (or make sense of) a lot of information before any of it is overtly used"* (*Blandford and Attfield 2010: 30*).

1.5.2 Shneiderman's "Framework for Mega-Creativity"

The *"Framework for Mega-Creativity"* by Shneiderman intends to support the design of powerful tools that can facilitate creative work by many people (*Shneiderman 2002: 214*). It builds on four activities:

- *Collect* (e.g., searching and browsing digital libraries, visualizing data and processes)
- *Relate* (e.g., consulting with peers and mentors)
- *Create* (e.g., thinking by free association, exploring solutions with "what-if" tools, composing artifacts and performances, reviewing and replaying session histories)
- *Donate* (disseminating results)

Similar to the Information Journey, these four activities are not a linear path, since creative work may require to return to earlier phases and much iteration (*Shneiderman 2002: 214*).

1.5.3 Lehikoinen et al.'s "GEMS" Framework

The GEMS framework is a high-level framework for understanding personal content experience and describes a lifecycle of personal content usage. It intends to consider the human perspective of how users interact with personal content, and what actions are performed on it. The framework is based on four phases (*Lehikoinen, Aaltonen, Huuskonen et al. 2007: 73*):

- *Get* – Users obtain the content from somewhere (e.g., receives, creates, captures, purchases).
- *Enjoy* – Users view, read, or listen to the content. They edit, remix, or personalize it.
- *Maintain* – Users maintain the content by organizing, archiving, rating it.

- *Share* – Users share the content by publishing, giving, sending, trading, or printing it.

Again, these phases are not sequential. Not all pieces of content go through all the steps but may jump from any phase to any other, e.g., they can be enjoyed after maintaining them (*Lehikoinen, Aaltonen, Huuskonen et al. 2007: 75*). Furthermore, all phases include activities of *searching* and *browsing* personal content. Both are essential for every phase.

1.5.4 Card's "Knowledge Crystallization"

In (*Card 2008*), Card introduces his framework of *"knowledge crystallization"*. In knowledge crystallization tasks, there is a goal (sometimes ill-structured) that requires the acquisition and making sense of a body of information, as well as the creative formulation of a knowledge product, decision, or action (*Card 2008: 540*). Card gives examples such as writing a scientific paper, business or military intelligence, weather forecasting, or buying a laptop computer. For such tasks, Card identifies four *knowledge crystallization operators* describing typical activities:

- *Acquire information* (e.g., monitor, search, capture, make implicit knowledge explicit).

  Examples: 1.) A users starts with an overview of films, and then uses sliders to filter them by metadata criteria such as year, length, or actors. 2.) A user uses a chart to visualize data about hundreds of stocks and industries and notices interesting trends among them.

- *Make sense of it* (e.g., extract information, fuse different sources, find schema, recode information into schema).

  Examples: 1.) To increase her return, a hotel manager extracts information from hotel occupancy data by visualizing it in different permutation matrices to reveal periodic patters. 2.) A customer extracts information from different sources about features of laptop computers to get an overview of available models.

- *Create something new* (e.g., organize for creation, author).

  Examples: 1.) A hotel manager authors a simplified diagram for a presentation of main findings, e.g., that a December convention does not seem to have effect of the other conventions to bring in guests. 2.) A customer creates a table of features by model as a compact description to facilitate comparison and decision making.

- *Act on it* (e.g., distribute, apply, act).

  Examples: 1.) A user distributes a report or gives a briefing. 2.) A user buys a laptop computer.

1.5.5 Knowledge Work in this Thesis

Although very different in their context and level of abstraction, all these frameworks have characteristic commonalities. For the purpose of this thesis, I therefore provide a operational definition of knowledge work based on these commonalities:

The goal of knowledge work is to *act on* the results of a knowledge-intensive process, e.g., to *make a decision* (*Blandford and Attfield 2010: 30*), to *apply new knowledge* (*Card 2008: 540*), or to *share* or *disseminate* a newly created information item (*Lehikoinen, Aaltonen, Huuskonen et al. 2007: 73*; *Shneiderman 2002: 214*).

Typically, after *exploring solutions* and using *what-if tools,* this item is created by *composing artifacts and performances* (*Shneiderman 2002: 214*), *editing* or *remixing* content (*Lehikoinen, Aaltonen, Huuskonen et al. 2007: 73*), *authoring presentations* (*Card 2008: 540*), or *writing articles* (*Blandford and Attfield 2010: 30*).

To achieve results, knowledge work involves a phase of *getting*, *acquiring*, or *collecting* information from one or many information sources. This involves activities of *searching*, *browsing*, and *visualizing* (meta-)*data* or *processes* (*Shneiderman 2002: 214*) to *make implicit knowledge explicit* (*Card 2008: 540*).

It is also necessary that the actual information items can be evaluated or *enjoyed* by *viewing* or *listening to* them (*Lehikoinen, Aaltonen, Huuskonen et al. 2007: 73*).

Knowledge work involves a phase of *interpreting*, *validating* (*Blandford and Attfield 2010: 30*), and *making sense* of information (*Card 2008: 540*). This involves *extracting information*, *fusing different sources*, and *finding a new schema to recode information* in it (*Card 2008: 540*). This can be done by *organizing*, *archiving*, or *rating* items (*Lehikoinen, Aaltonen, Huuskonen et al. 2007: 73*). According to (*Kidd 1994*) and (*Andrews, Endert, and North 2010*), altering the *spatial layout* of information items and the *spatial relations* between them is a particularly important part of such sensemaking during knowledge work. The same is true for leaving *marks*, *traces*, or *annotations* on items.

Knowledge work consists of different phases of higher level activities that are separated in time and are executed either sequentially or in random order depending on the context, goals, and progress. Each higher level activity, e.g., *Collect* (*Shneiderman 2002: 214*), or *Acquire Information* (*Card 2008: 540*), is comprised of smaller tasks, e.g., *searching and browsing,* or *capture*, that can be supported by an interactive system.

Since true knowledge work cannot be automated, an interactive system cannot "understand" the information it is holding or predict what the user wants to do with it (*Kidd 1994*). A user interface cannot provide the structure that is lacking in the task itself. The interface can, however, provide a context for interaction to guide the user through a space of possible actions and results (*Collins 1995: 31*).

A brief example from this thesis can illustrate how collaborative knowledge work can be supported in practice: the *Media Seminar Room* prototype in section 2.1 (p.30) is designed for letting multiple students of media science collaborate in an interactive space during a seminar about a certain era of cinema. The prototype provides a shared visual workspace with an overview of all relevant movies clustered by genres on a large vertical wall display and on a horizontal interactive tabletop.

By zooming into the different movie objects on the wall display or tabletop, the students can access all the data about the movies, their plot, directors, cast, etc., and they can watch the movie itself as a video stream. Depending on their task, they can add virtual Post-It notes to the overview as annotation. Furthermore, they can rearrange the spatial configuration of movies, e.g., to select a few movies to work on or to establish a new spatial schema that is not based on the genre, but on a different angle that is related to the topic of their seminar (e.g., plot, cast, director, production year).

By providing multiple devices that can access individual regions of the shared visual workspace, the interactive space also supports phases of loosely-coupled parallel work by individual group members. This enables a distributed process of sensemaking that can lead to formulating different hypotheses and to elaborate on them collaboratively in a term paper or seminar presentation. This final step of creating a new knowledge artifact and acting on it concludes the process of collaborative knowledge work.

1.5.6 Support of Knowledge Work in this Thesis

It is important to notice that the interactive spaces in this thesis do not support all activities of knowledge work to the full extent at the same time. Instead, they focus on different selected activities of knowledge work and illustrate possible solutions for design and implementation. For example, in the *Media Seminar Room*, the actual word processing of the term paper and the authoring of the presentation are not supported by the interactive space but are done using ordinary desktop or laptop PCs outside the interactive space. The prototype's design and implementation are instead focused on the way that movies can be visualized, accessed, reorganized, and annotated by users.

Another example is the prototype (or apparatus) for the study of spatial memory and navigation performance in section 4.5 (p.139). It was designed and implemented to study an important but also much focused lower level single-user interaction within knowledge work: spatial navigation to and memorization of object locations in a zoomable visual workspace. While this is only a small part of the higher level activities of collaborative knowledge work, understanding the underlying cognitive mechanisms is of great importance for the design of zoomable user interfaces for interactive spaces and is therefore the main focus of that prototype.

1.6 Research Goal

As introduced at the beginning of this chapter, the research question for this thesis is:

> *How can designers and developers of ubiquitous computing environments be supported to create more usable multi-user and multi-device post-WIMP interactive spaces for co-located collaborative knowledge work?*

The previous sections have clarified what kind of ubiquitous computing environments, interactive spaces, devices, and collaboration the research question refers to. The following sections clarify the *research goal* of this thesis by describing what means of *supporting designers* and *supporting developers* are intended as an outcome.

1.6.1 Supporting Designers

In user interface and interaction design, there is a long tradition of providing golden rules, principles, guidelines, or heuristics to designers to let them benefit from the insights of other designers or researchers. Famous examples are Shneiderman's *"Eight Golden Rules of Interface Design"* (*Shneiderman 1997*), Norman's *"Seven Principles of Design"* (*Norman 2002*), or Nielsen's *"Ten Usability Heuristics"*[8], but other guidelines were published already in the early 1980s (*Malone 1982*) and even 1970s (*Hansen 1971*). More recently, researchers have started to formulate design principles also for post-WIMP UIs. For example, Dourish has formulated design principles to point out a set of *"things to pay attention to"* when designing post-WIMP embodied interaction (*Dourish 2004: 160*).

Following this tradition, this thesis sets out to formulate a *set of design principles* for the design of post-WIMP interactive spaces for collaborative knowledge work. They are intended to remind designers of important properties that these user interfaces should expose and they suggest the interaction and visualization techniques to achieve them. This set of design principles is derived from scientific or professional literature from the fields of HCI, Cognitive Science, CSCW, and InfoVis. It is based on theories, models, and frameworks from these disciplines, but also on selected designs from scientific publications and the findings from their evaluation in user studies. Thus the first step of supporting designers in this work is to compile and select relevant models and designs. In a second step, deduction and logical reasoning are used to relate and fuse different designs or sources of information and to discuss their origin, differences, and commonalities. Eventually, in a third step, this in-depth discussion arrives at the formulation of new design principles as a new scientific contribution. This is similar to the approach chosen by more theoretical and conceptual work in HCI, e.g., Dourish's "Where the Action is" (*Dourish 2004*) or Jacob et al.'s Reality-Based Interaction (*Jacob, Girouard, Hirshfield et al. 2008*).

Since deduction in HCI is typically a matter of interpretation and argumentation, it cannot be executed with the same logical and mathematical rigor as in theoretical computer science or algorithms. Therefore HCI has a strong tradition of observation and experimentation to empirically validate the claims deduced from theory. To enable such empirical research, this thesis applies the suggested design principles in practice and applies them to create novel artifacts such as new *prototypes* for empirical observation and experimentation. This is the second contribution of this thesis and consists of two parts: First, the resulting new artifacts (e.g., the design of *DeskPiles* of *Facet-Streams* in chapter 2) serve as examples to illustrate a design principle and as sources of inspiration for other designers, researchers, and practitioners. Second, the empirical studies of the suggested design approaches with student designers (see section 3.2) or the evaluation of resulting *prototypes* with end-users (see chapter 6) contribute to the validation of the design principles and enable a deeper understanding of their cognitive basis.

[8] Ten Usability Heuristics. *http://www.useit.com/papers/heuristic/heuristic_list.html* (Accessed Jul 5, 2012)

1.6.2 Supporting Developers

In 2000, Myers et al. discussed a future of ubiquitous computing in which new form factors like mobile phones, PDAs, or wall-size displays will lead to a simultaneous use of multiple devices that will also replace the *"standard desktop model"* of GUIs. They anticipated, that this will also affect UI developers and will create dramatic new needs for tools to build those interfaces (*B. Myers, Hudson, and Pausch 2000*). For example, Myers et al. mentioned the need to support *"varying input and output capabilities"* such as different screen sizes and resolutions and *"tools for coordinating multiple, distributed communicating devices"*.

Today, developers of post-WIMP user interfaces face many of these predicted challenges. This becomes obvious when looking at the ongoing research on tools and architectures for implementing post-WIMP multi-user, multi-modal, or multi-device user interfaces, e.g., (*Gjerlufsen, Klokmose, Eagan et al. 2011*; *Johanson, Fox, and Winograd 2002*; *Kim, Javed, Williams et al. 2010*; *Klokmose and Beaudouin-Lafon 2009*; *W. König, Rädle, and Reiterer 2010*; *Streitz, Tandler, Müller-Tomfelde et al. 2001*). Furthermore, Shaer and Jacob discuss very similar challenges for Tangible User Interfaces (TUIs). TUI developers face difficulties such as *"the lack of appropriate interaction abstractions"*, *"the shortcoming of current software tools to address continuous and parallel interactions"* and *"the excessive effort required to integrate novel input and output technologies"* (*Shaer and Jacob 2009*).

A common approach to facilitate UI implementation for developers is the creation of software libraries, application programming interfaces (APIs), or software frameworks as *"user interface software tools"* (*B. Myers, Hudson, and Pausch 2000*). They provide software components that contain the functionality and algorithms that are typically needed by developers of user interfaces. This enables developers to reuse established and proven off-the-shelf components instead of having to implement the entire lower level functionality or underlying algorithms themselves. It also enables developers to approach implementation based on higher level abstractions and concepts (e.g., UI objects, controls, visualizations of data, multi-touch manipulations) that are closer to the intended interactive behavior and the developers' mental model of the UI than the underlying details of implementation (e.g., pixels, variables, methods, classes, input events). *"In general, tools help to reduce the amount of code that programmers need to produce when creating a user interface, and they allow user interfaces to be created more quickly. This, in turn, enables more rapid prototyping and, therefore, more iterations of iterative design that is a crucial component of achieving high-quality user interfaces (…) Tools influence the kinds of user interfaces that can be created. Successful tools use this to their advantage, leading implementers toward doing the right things, and away from doing the wrong things"* (*B. Myers, Hudson, and Pausch 2000*).

As a consequence, similar to HCI's tradition of providing design principles to designers, there is also a tradition of providing software frameworks to UI developers. There are several examples of successful UI frameworks from academic research or the open source community, e.g., for zoomable user interfaces (*Bederson, Grosjean, and Meyer*

2004), tabletops (*Shen, Vernier, Forlines et al. 2004*), information visualization (*Heer, Card, and Landay 2005*), or vision-based multi-touch UIs[9].

Following this tradition, this thesis provides a *software framework* that facilitates the implementation of user interfaces that are designed following the above-mentioned *set of design principles*. It provides important core functionality to realize such designs, e.g., object-oriented and zoomable user interfaces that are distributed across multiple devices such as wall-sized displays, interactive tabletops, or tablet PCs. This software framework is shared with other researchers, developers, and practitioners as open source[10] and is described in great detail in section 3.4 of chapter 3.

To validate the practical value of the software framework, it was used for developing various prototypes that are described in this thesis. Furthermore, two user studies of its API usability with student developers were conducted and are described in section 3.5. An indicator for the practical value and relevance of the framework is that during the course of my PhD project many other researchers from our group have already used the framework to prototype and study novel user interfaces in the context of their research on interactive spaces, Reality-Based Interaction, and collaborative interfaces for search or creative design, e.g., (*Demarmels, Huber, and Heilig 2010*; *Geyer and Reiterer 2010*; *Geyer, Pfeil, Budzinski et al. 2011*; *Geyer, Pfeil, Höchtl et al. 2011*; *Geyer, Budzinski, and Reiterer 2012*; *Heilig, Demarmels, Rexhausen et al. 2009*; *Heilig, Demarmels, Allmendinger et al. 2010*; *Heilig, Huber, Gerken et al. 2011*; *Heilig 2012*; *Jenabi 2011*).

1.6.3 The ZOIL Paradigm

In summary, this thesis answers the research question by providing designers with deriving and presenting the afore-mentioned *set of design principles*. These principles are further illustrated by introducing new *prototypes*. Selected *prototypes* are subject to empirical studies to learn more about the appropriateness of their designs in practice. Furthermore, to support the developers, this thesis provides a new *software framework* that facilitates the implementation of user interfaces that are designed according to the suggested design principles.

In analogy to Thomas Kuhn's concept of a *scientific paradigm (Kuhn 1962),* the entirety of these contributions constitutes what economist Giovanni Dosi refers to as a "*technological paradigm" (Dosi 1988)*: *'A "technological paradigm" defines contextually the needs that are meant to be fulfilled, the scientific principles utilized for the task, the material technology to be used. In other words, a technological paradigm can be defined as a "pattern" of solution of selected technoeconomic problems based on highly selected principles derived from the natural sciences (…) A technological paradigm is both an exemplar – an artifact that is to be developed and improved (…) – and a set of heuristics (…)"* (*Dosi 1988*).

[9] Touchlib, *http://www.nuigroup.com/touchlib/* (Accessed Jul 8, 2012)

[10] ZOIL Software Framework. *http://zoil.codeplex.com/* (Accessed Jul 10, 2012)

Dosi's emphasis on materials and natural sciences results from the focus of Dosi's original work on engineering problems like manufacturing cars or integrated circuits. Since this thesis treats problems of HCI and designing and implementing interactive products, the equivalents to Dosi's *"material technology"* and *"exemplars"* are the prototypes and the software framework presented in this thesis. The equivalents to Dosi's *"principles derived from the natural sciences"* are the presented design principles and theoretical models which are derived from HCI, CSCW, InfoVis, and Cognitive Science literature and the findings from the conducted empirical studies.

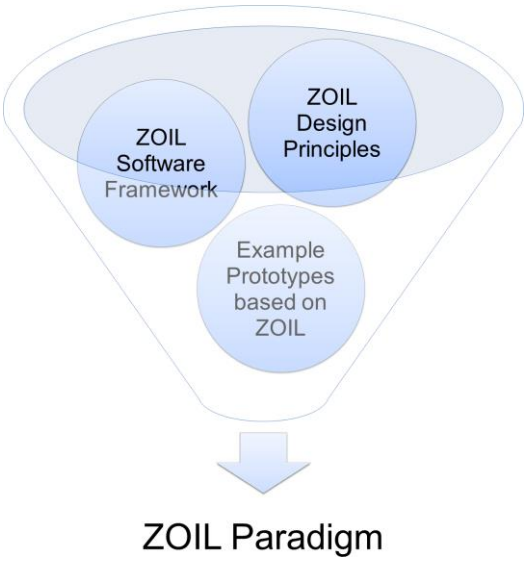


Figure 9 – Visual illustration of the ZOIL paradigm and its components.

As a consequence, the sum of the contributions of this thesis formulates a new technological paradigm that I henceforth refer to as the *ZOIL paradigm* (Figure 9). Similar to WIMP, ZOIL is an acronym that describes the components of a user interface and the defining characteristics of the interaction with it: **Z**oomable **O**bject-Oriented **I**nformation **L**andscape. In a ZOIL-based user interface, all kinds of information items and tools are organized in a continuous information landscape in space and scale that serves as zoomable workspace. The user interface is object-oriented without identifiable application boundaries and distributed across device boundaries.

ZOIL-based user interfaces can be described in terms of the underlying six *ZOIL design principles*[11] that are introduced in chapters 3 to 6:

1. Provide post-WIMP functionality as objects, not applications. (chapter 3)
2. Provide a zoomable user interface for navigation with semantic zooming. (chapter 4)
3. Provide space for sensemaking, marks, and annotations. (chapter 5)

[11] During my PhD project, the initial formulation of the ZOIL design principles changed many times. This is also reflected in my various publications on ZOIL. The exact formulation, wording, and order changed and also the total number of design principles grew from 4 to 6. The formulation, wording, and order of the six ZOIL design principles in this thesis should be considered as the most recent by the reader.

4. Provide space for coordinating mixed-focus collaboration. (chapter 5)
5. Provide post-WIMP InfoVis tools for fluid interaction. (chapter 6)
6. Support multi-user collaboration with visual-tangible externalizations. (chapter 6)

The technological counterpart to the *ZOIL design principles* is the *ZOIL software framework* that provides the necessary functionality to implement user interfaces that follow these design principles. For the remainder of this thesis, I refer to the entirety of contributions including the design principles, design cases, prototypes, study results, and the software framework as the *ZOIL paradigm* or just *ZOIL*.

1.7 Research Scope

This section further refines the formulation of the research question and goal for this thesis by explicitly including or excluding certain scientific and technological challenges.

1.7.1 Employed Technology

In accordance to (*Bell and Dourish 2006*), all *prototypes* and the *software framework* that were created as part of this thesis are rooted in present-day ubicomp technology and do not speculate on a proximate future in which current problems of consistency, standardization, sensor technology, or performance are already solved. They are based on existing enabling technologies such as commercially available devices (e.g., Microsoft Surface tabletop computers, Nintendo Wiimote controllers), programming platforms (Microsoft .NET 4.0, Windows Presentation Foundation, and the C# programming language), communication protocols (TCP/IP, Open Sound Control) and operating systems (Windows Vista, Windows 7). The design and development of new ubicomp hardware, communication protocols, or discovery mechanisms is not part of this work. By using today's commercially available hardware, popular network protocols, and operating systems and by sharing the underlying software framework as open-source, the results of this thesis can be easily reused by other researchers and practitioners.

1.7.2 Targeted User Groups

The targeted end-user groups vary between the different prototypes presented in this thesis. While the *EuroITV* application (see section 4.3.2) or *Facet-Streams* (see section 2.4) are intended to be used by novice users in an entertainment or retail context after no or only few minutes of training, *DeskPiles* (see section 2.2) is designed for scientists with extensive skills of using, customizing and reappropriating existing computing technologies during their daily work.

A further targeted user group are interaction designers and software developers concerned with the design and implementation of post-WIMP interactive spaces. The case study and the API usability evaluations of the ZOIL software framework in chapter 3 focused on undergraduate and graduate students of computer science. While some had no prior experience with the C# programming language or the .NET 4.0 and Windows Presentation Foundation (WPF) platform, all of them had prior experience with object-

oriented programming of GUIs for WIMP applications. Evaluations with professional designers or developers from industry are not part of this work.

1.7.3 Design of the Physical Environment and Form Factors

Although this thesis is about interactive *spaces* in which users and devices are co-located in rooms such as meeting rooms or design studios, the architectural layout and physical ergonomics of such rooms, devices, or interactive furniture are not part of this work. This thesis focuses on the interaction design and ergonomics of the software side of interactive spaces and touches physical ergonomics only very briefly in context of small tangible user interface elements, e.g., in *Facet-Streams* (see section 6.3.3), or in studying the effect of proprioception and kinesthesia on spatial memory (see section 4.5). Furthermore this thesis does not give any recommendations how to integrate such interactive spaces in existing architectural contexts or organizational workflows.

1.7.4 Multi-Display Environments

In HCI, there is an existing body of knowledge about the design and implementation of multi-display environments (MDEs), e.g., (*Balakrishnan and Baudisch 2009*; *Terrenghi, Quigley, and Dix 2009*). Although this thesis touches some streams of MDE research, it is not focused primarily on MDEs and explicitly excludes some of their typical challenges.

For example, one stream of MDE research is closely related to the design of the physical environment and form factors, e.g., the effect of display size and curvature on user performance and insights (*Shupp, Andrews, Dickey-Kurdziolek et al. 2009*). As already mentioned in the previous section, such considerations of physical design and form factors are not part of this thesis.

Other typical MDE research questions are how different types of displays impact users and human collaboration. For example, researchers study the effect of display configuration and usage context on people's behavior in group situations, e.g., social anxiety and willingness to change answers (*Robles, Nass, and Kahn 2009*) or more equitable participation (*Rogers, Lim, Hazlewood et al. 2009*). Some own work related to this can be found in this thesis in chapter 6 where the effect of visual-tangible externalizations on a horizontal tabletop is discussed in comparison to a traditional vertical display with a Web interface. However, this thesis does not formulate or validate generic models for such effects, nor does it try to give guidelines for MDE configurations that achieve better collaboration.

Another frequently addressed challenge is how to make MDEs work more seamlessly (*Balakrishnan and Baudisch 2009*), for example by introducing techniques of cross-display object movement (*Nacenta, Gutwin, Aliakseyeu et al. 2009*) or user interfaces for relocating applications (*Biehl and Bailey 2004*). An important part of such research is the position tracking of users, displays, and devices in three-dimensional space to enable a natural movement of objects across displays and devices using pointing or gestures. Such cross-display object movement techniques are not part of this work. However, my fellow PhD candidate Mahsa Jenabi has used the ZOIL software framework to create and

evaluate such an MDE with mobile devices (*Jenabi 2011*). Furthermore, the Master's thesis of Simon Fäh has also used the ZOIL software framework to explore bi-manual pointing gestures for cross-display object movement (*Fäh 2011*).

1.8 Organizational Overview

A typical structure of a PhD thesis in HCI is a chronological presentation of several prototypes created during the PhD project and findings from their evaluation with users. The reoccurring themes observed during the user studies are collected and structured in a conceptual framework that is then presented in the last chapter as scientific outcome and conclusion.

During structuring this thesis, I decided to put a strong emphasis on this thesis' outcome, i.e., the ZOIL paradigm with its design principles. Therefore I decided for a different organizational structure: Each of the following chapters focuses on one or two ZOIL design principles as an outcome and how they were applied, implemented, and/or evaluated in the ZOIL example prototypes or the ZOIL software framework.

Chapter 2. The goal of this chapter is to make the reader familiar with the general interaction and visualization style in ZOIL-based interactive spaces. Therefore it illustrates selected example prototypes from a users' perspective by presenting typical scenarios of use and visual material such as sketches, photographs, and screenshots. It concludes with a table summarizing the different ZOIL example prototypes and how they relate to the ZOIL design principles that are introduced in chapters 3-6.

Chapter 3. This chapter introduces the concept of *Object-Oriented User Interfaces* (OOUIs) and applies it on post-WIMP interaction in interactive spaces. Within ZOIL, *object-oriented* instead of *application-oriented* interaction is important for achieving a seamless interaction with different information types and tools.

The first part of this chapter is about understanding and designing OOUIs. It revisits and summarizes different views of OOUIs in HCI literature of the WIMP era and enters new terrain by applying them on post-WIMP interaction. The first part of this chapter concludes with the formulation of the *1st ZOIL design principle*.

The second part of this chapter is about implementing OOUIs for post-WIMP interactive spaces in multi-user, multi-display, and multi-device settings. For this purpose, it introduces the *ZOIL software framework* that serves as a kind of middleware between the application level and the operating system and provides high-level functionality in the areas of presentation & interaction, network communication, and persistence & synchronization. The second part concludes with two evaluation studies of the ZOIL software framework's API usability with student designers and developers and a discussion of the framework's practical value.

Chapter 4. This chapter introduces *Zoomable User Interfaces* (ZUIs) and discusses their key role within the ZOIL paradigm. ZOIL uses ZUI principles to replace traditional concepts of the WIMP desktop metaphor such as files, folders, and application windows.

Instead, ZOIL introduces a zoomable workspace (the 'information landscape') as a model-world UI for natural and consistent management of digital information items and interaction with virtual tools.

In a first step, this chapter discusses ZUI history and the foundations and mathematics of ZUI interaction. Then it presents ZUI examples taken from different ZOIL-based prototypes and gives an overview about the implementation of ZUIs in and with the ZOIL software framework. It continues with a user study of the effect of multi-touch interaction on ZUI navigation and spatial memory. It concludes with the formulation of the *2nd ZOIL design principle*.

Chapter 5. This chapter introduces and discusses the key role that virtual and physical space play for the design of ZOIL-based user interfaces. Based on empirical studies from HCI and cognitive science, this chapter argues for considering space an integral part of human cognition and a key resource for collaborative knowledge work.

In the first part of this chapter, ZOIL's information landscape is used as a virtual space for the purpose of *sensemaking* and to enable *marks* and *annotations*. Different uses of space in ZOIL example prototypes are illustrated and results of a user study of the *DeskPiles* prototype are presented that reveal how space and annotation is used in realistic usage situations. This part concludes with the formulation of the *3rd ZOIL design principle*.

The second part of this chapter discusses the role of space for collaboration by first looking at how users partition physical space into territories to coordinate their collaboration at interactive tabletops. Then this is applied to collaboration in virtual workspaces in order to support many different coupling styles and fluid transitions between them during *mixed-focus collaboration*. This part concludes with the formulation of the *4th ZOIL design principle*.

Chapter 6. This chapter briefly introduces the fundamentals of information visualization and how it can help users to manage today's growing number of functions and information items. A summary of relevant cognitive models explains how post-WIMP interaction styles using multi-touch and tangible input can achieve an enhanced user experience of information visualization with a *fluid interaction*. This is followed by an in-depth description of the design and user studies of *Facet-Streams* which serves as a *"best-in-class"* example of post-WIMP and fluid information visualization. This chapter concludes with formulating the *5th and 6th ZOIL design principle.*

Chapter 7. This chapter concludes this thesis by summarizing its contributions and giving an outlook on future work.

2 Examples of ZOIL-based Interactive Spaces

The goal of this chapter is to make the reader familiar with the general interaction and visualization style in ZOIL-based interactive spaces. Therefore it illustrates selected ZOIL-based example prototypes from a users' perspective by presenting typical scenarios of use and visual material such as sketches, photographs, and screenshots.

This chapter aims at giving the reader a representative impression of ZOIL-based interaction before the following chapters introduce the underlying design principles and a more precise terminology and also discuss the underlying scientific background and the employed technology in detail. Furthermore, the presented prototypes and their design in this chapter can serve as a source of inspiration to other researchers or designers. For this reason, this chapter also uses some figures that reappear in the following chapters. This redundancy is intentional to achieve better readability and comprehensibility.

All examples in this chapter were designed following the ZOIL design principles and have been implemented using the ZOIL software framework.

The first ZOIL example prototype is the *Media Seminar Room*. It was conceived of, designed, and implemented by me as part of my PhD project with support from Mathias Heilig and a team of student developers (Michael Zöllner, Mischa Demarmels, Stephan Huber, Oliver Runge, Benjamin Frantzen, Sebastian Rexhausen) to demonstrate the functionality of my ZOIL software framework. The *Media Seminar Room* has not been published previously.

The second ZOIL example prototype is *DeskPiles*. It is the result of joint work with Natasa Milic-Frayling of Microsoft Research Cambridge, and Jeremy Baumberg of the NanoPhotonics Centre of the University of Cambridge, where the prototype was installed to explore novel ways of information management for e-Science. *DeskPiles* was conceived of and designed by me together with Master's student Toni Schmidt with whom I also implemented the prototype. Additional design and implementation was provided by Jens Gerken and Michael Zöllner. *DeskPiles* has not been published previously as a part of a PhD, Bachelor's or Master's thesis, but in a technical project report (see section 2.2, p.37). Images of it have previously appeared in (*Jetter, Zöllner, and Reiterer 2011*; *Jetter, Zöllner, Gerken et al. 2012*; *Seifried, Jetter, Haller et al. 2011*).

The third ZOIL example prototype is *Distributed Sketching*. The prototype is based on the *Media Seminar Room* and is joint work with Florian Geyer who applies my ZOIL design principles and my ZOIL software framework in his PhD project to create different interactive spaces for creative designers. The prototype was designed by Florian Geyer

and implemented by his team of student developers (Jochen Budzinski, Anita Höchtl, Markus Hankh). It has been previously published as a poster in (*Geyer, Jetter, Pfeil et al. 2010*) and images of it have appeared in (*Reiterer 2011*).

The fourth ZOIL example prototype is *Facet-Streams*. It is the result of joint work with Natasa-Milic Frayling of Microsoft Research Cambridge on collaborative faceted search on tabletops. *Facet-Streams* was conceived of and designed by me together with Jens Gerken and previously published in (*Jetter, Gerken, Zöllner et al. 2010a*; *Jetter, Gerken, Zöllner et al. 2011*). The technical implementation was done by me with support from student developer Michael Zöllner. Section 2.4 contains a description of *Facet-Streams* that I wrote for a journal article about fluid interaction for information visualization (*Elmqvist, Vande Moere, Jetter et al. 2011*).

2.1 The Media Seminar Room

The *Media Seminar Room*[12] prototype is designed for letting multiple students of media science collaborate in an interactive space during a seminar about films, e.g., about films of a certain era or by a certain director. For this reason, the system provides the students with a database of selected films. Similar to the *Internet Movie Database (IMDb)*, each entry for a film contains detailed information about the plot, the cast, etc. and is not only described textually but also contains posters and the video stream of the film. The goal of the *Media Seminar Room* is to let students make sense of this information by accessing, organizing, and annotating this information in space and scale. During collaborative search, discussion, and annotation, students create the ideas and hypotheses that they later elaborate in term papers or seminar presentations.

Physically, the *Media Seminar Room* consists of two large back projected display cubes as vertical screens for pen input and a Microsoft Surface interactive tabletop for multi-touch input and for tracking physical objects (Figure 10).

After startup, the prototype provides a shared visual workspace, i.e., ZOIL's information landscape, on the two vertical screens and on the tabletop simultaneously. The information landscape contains a visual overview of all movies for the seminar that consists of several genre clusters of films (Figure 11). Users can zoom and pan in the information landscape that contains the overview. By zooming in, the individual movie objects grow in display space and their visual representation changes and details about the movie (e.g., cast) and the video stream of the movie become visible (see Figure 12 and section *4.3.4 Semantic Zooming,* p.125).

Students can gather around the tabletop to collaboratively explore and change the initial overview of films in the information landscape. At any time, they can rearrange the spatial configuration of movies using multi-touch manipulations, e.g., they can drag movies to a different location and resize or rotate movies. Furthermore, since the

[12] Video containing some features of *Media Seminar Room*: *http://www.youtube.com/watch?v=Np1ODKU48do* (Accessed Jul 14, 2012)

information landscape is zoomable, objects can not only be organized in space, but also in scale, for example to express different priorities or groupings.

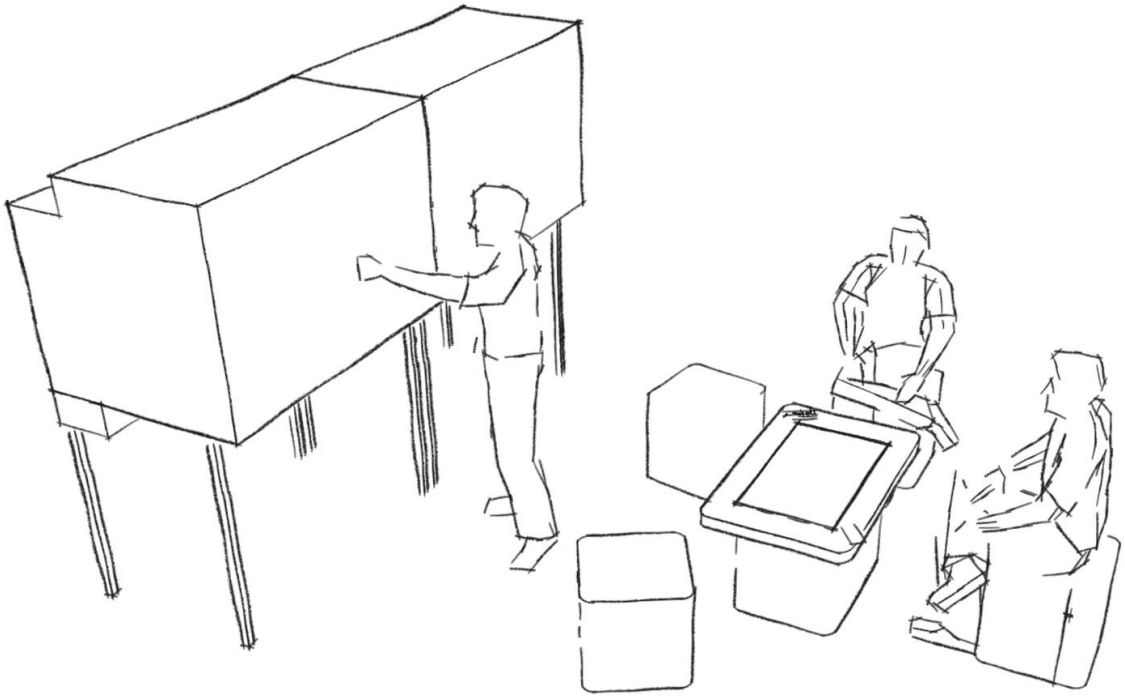


Figure 10 – A sketch of the Media Seminar Room. It contains two 67" back projected vertical screens with pen-input and a 30" Microsoft Surface tabletop with multi-touch and tangible input.

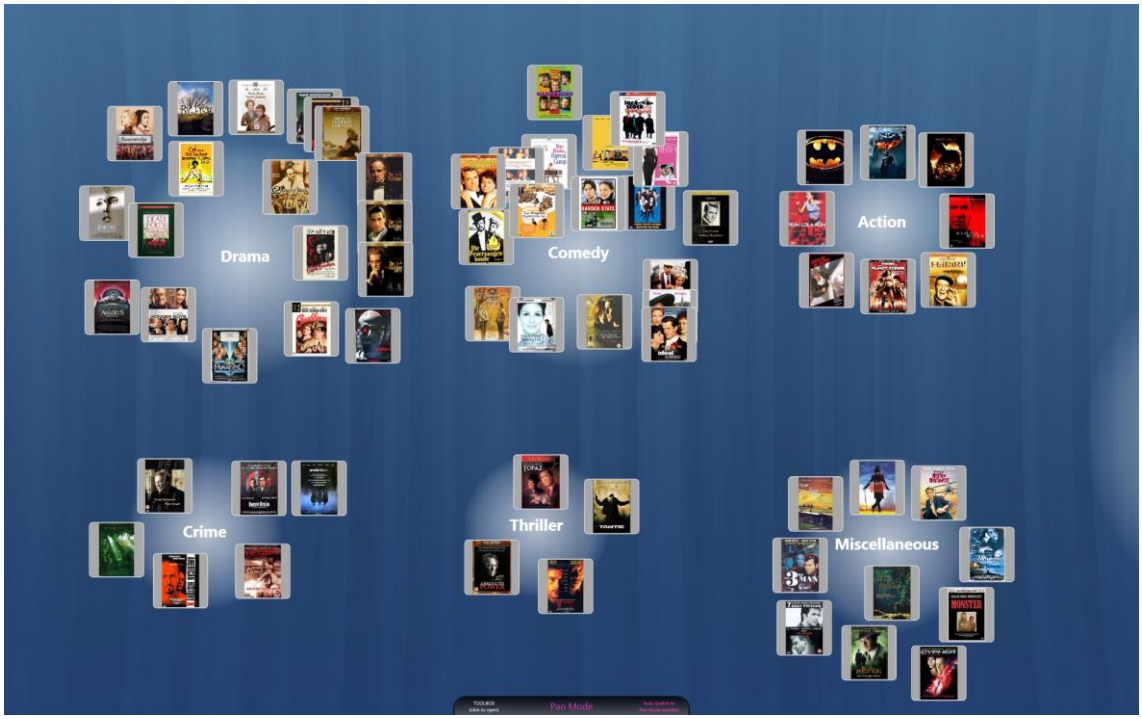


Figure 11 – ZOIL's information landscape contains movie objects clustered by genre.

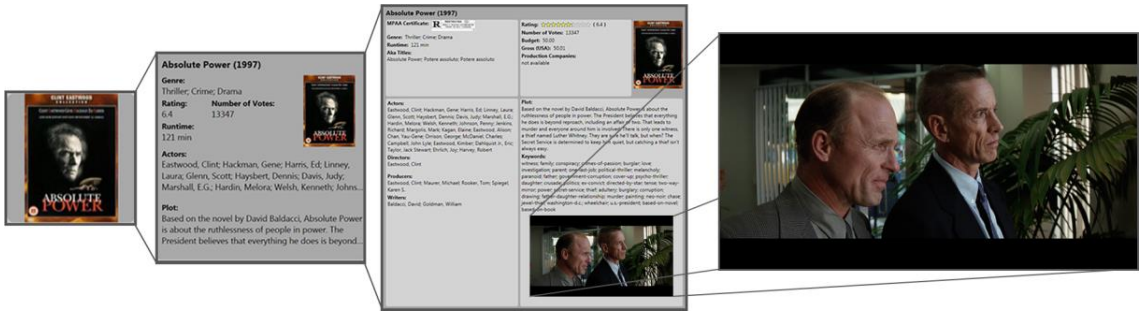


Figure 12 – Zooming into a film object can be used to access its details and also its video stream.

By exploration and manipulation of the initial overview, students can select the relevant movies from the collection to work on or they can establish a new spatial schema that is not based on the genre, but on a different 'angle' that is related to the topic of their seminar (e.g., plot, cast, stylistic elements).

Figure 13 – A tabletop computer can be used to collaboratively explore and rearrange the movie objects in ZOIL's zoomable information landscape.

As described, the *Media Seminar Room* provides three interactive surfaces for accessing and manipulating the shared information landscape: the two vertical screens and the tabletop. Thereby each of these interactive surfaces (or active input/output devices) serves as a kind of camera that views a section of the information landscape from above and that can be individually controlled by using zooming and panning commands. Figure 14 shows an example information landscape containing a world map to illustrate

ZOIL's principle of a device as a camera: Each device 1-3 displays a different rectangular section 1-3 of the map. By using zooming commands, a device can enlarge the size of the visible region (zoom out) or shrink it (zoom in). By using panning commands (left/right, up/down), the visible region moves sideways in the map.

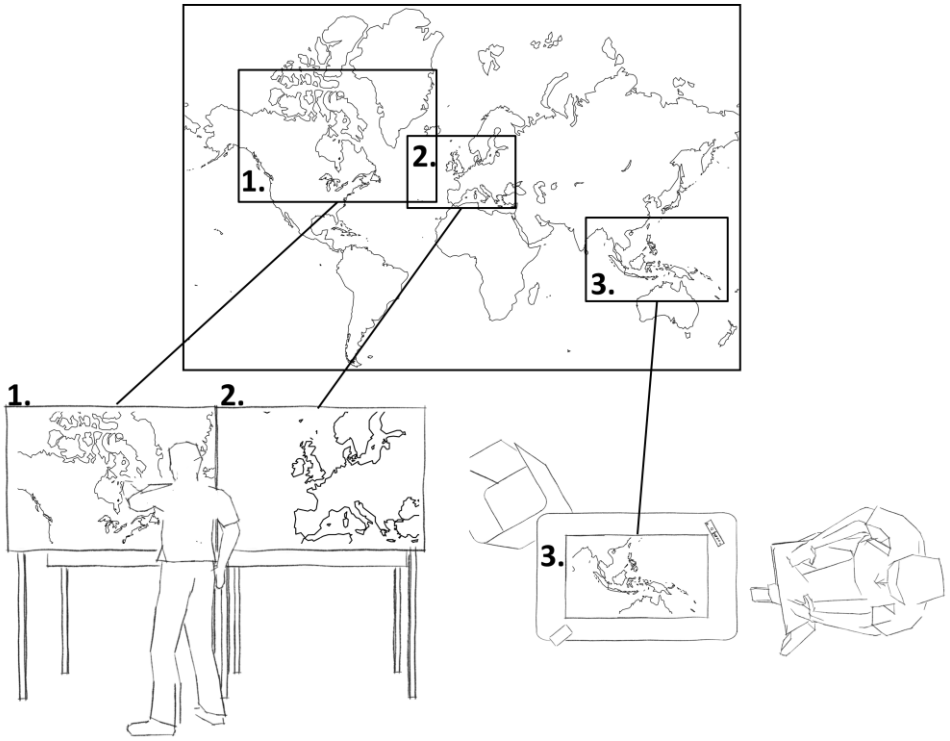


Figure 14 – The camera metaphor in a ZOIL-based interactive space. Each device 1-3 (bottom) shows a rectangular region of the shared visual workspace (top) that is controlled using zooming and panning.

Figure 15 – The right vertical screen shows the initial overview of films. The left screen shows the video stream of a film object after zooming into it.

While each device can be used as a camera providing an independent view, the changes that are made from a device within a visible region are immediately applied to the information landscape and are shared with all other devices (see sections *3.4.1 Implementing a Shared OOUI Object Space* and *3.4.3 Real-Time Synchronization*). This is comparable to a multi-player online game, where each player can move individually

inside of the virtual game world and has an individual view of it. However, all actions and modifications a player executes immediately change the state of the entire game world and thus affect all other players. In ZOIL, all changes to objects, their content, locations, sizes, or rotation angles are not only executed locally on one device but they are executed globally, so that the information landscape has always the same state on all devices in the interactive space regardless of their currently visible region.

ZOIL's concept of devices as cameras into the information landscape enables flexible styles of collaboration and coupling of displays (see chapter 5). For example, in the *Media Seminar Room*, users can start a session with zooming and panning on a vertical screen until it shows an overview of the entire workspace (Figure 15, right). While this overview stays on the vertical screen during the entire session, the tabletop is used to explore movie objects by frequently zooming in and out and rearranging them to form new clusters or piles of movies. If some of the details of a movie or a movie's video stream are of particular interest to the group, the second vertical screen can be used to display them (Figure 15, left) while preserving the overview on the first vertical display and continuing the rearrangement of objects on the tabletop.

To facilitate the assignment of different visible regions to the vertical displays, the *Media Seminar Room* also provides a physical object as a *camera lens* on the tabletop (see section *4.3.8 Tangible Lenses*). The lenses' position is continuously tracked by the vision system of the tabletop using fiducial optical markers that are sticking on the lenses' backside. By touching the "Cube 1" or "Cube 2" (Figure 16) icons next to the lower right corner of the lens, users can create a coupling between the lens and the left or right vertical display. When coupled, the remote vertical displays always render exactly the part of the information landscape that lies inside the lens' boundaries on the tabletop, but at a much higher resolution and detail. By moving the camera lens or by zooming and panning the workspace on the tabletop, this creates the illusion of moving a physical camera on the tabletop that remotely controls the content of the vertical display.

Figure 16 – The physical camera lens object on the tabletop can be used to establish coupling between the content that lies within the boundaries of the lens and the remote vertical displays.

A further use of physical lens objects on the tabletop, is the *visualization lens* that shows alternative renderings of the content inside the lenses' boundaries, e.g., lists, scatter plots, bar charts, or tables (see section *4.3.7 Lenses*). By moving the lens above a cluster of movies and touching the "Scatter Plot" icon next to the lower right corner of the lens (Figure 16), the underlying cluster is rendered as a zoomable scatter plot using metadata such as year and user rating on the X- and Y-axes to provide analytical overviews (Figure 17). This way, users can quickly identify the oldest and most popular movies in the top left corner or the most recent ones and unpopular ones in the bottom right corner. Thereby the scatter plot also uses the size-dependent representations from Figure 12 to reveal more details when zooming into the data point of an object.

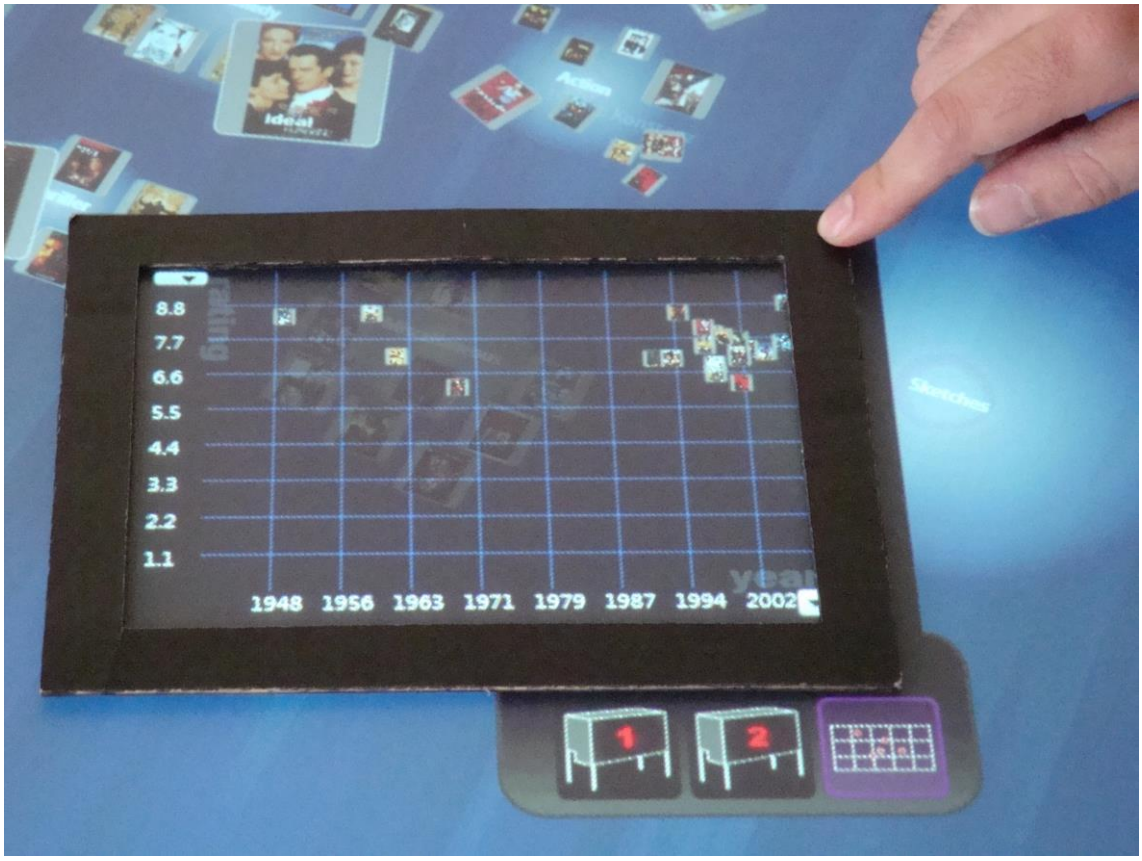


Figure 17 – A tangible lens object can be used to view the underlying objects in a scatter plot. In this case, year on the x-axis and user rating on the y-axis.

Since a physical lens object cannot be conveniently used on vertical displays and also has a static size and shape, there is also the alternative way of creating virtual lenses with touch and pen interaction. Users can use their fingers or a pen to draw arbitrary shapes around clusters to create a lens that selects the objects of interest. This selection is then displayed in a visualization next to the shape (Figure 18). The different visualizations that can be used inside the *Media Seminar Room* are shown in Figure 19.

During the collaborative process of sensemaking and discussion, the students might want to annotate particular interesting objects in the workspace or to keep track of their findings, new ideas, and hypotheses to elaborate them later in a term paper or seminar

presentation. For this purpose, virtual 'Post-It' notes can be added to the workspace. For example, the 'Post-It' lens on the tabletop in Figure 20 creates a virtual Post-It note at its position in the workspace. By writing on a physical piece of paper with an Anoto digital pen, all ink strokes from the paper are transmitted in real-time into its virtual counterpart. Additionally, the lens can also receive touch input for coarse-grained highlighting or drawing by finger (see section *4.3.8 Tangible Lenses*). On the vertical pen-enabled screens, it is also possible to use standard pen input to edit the content.

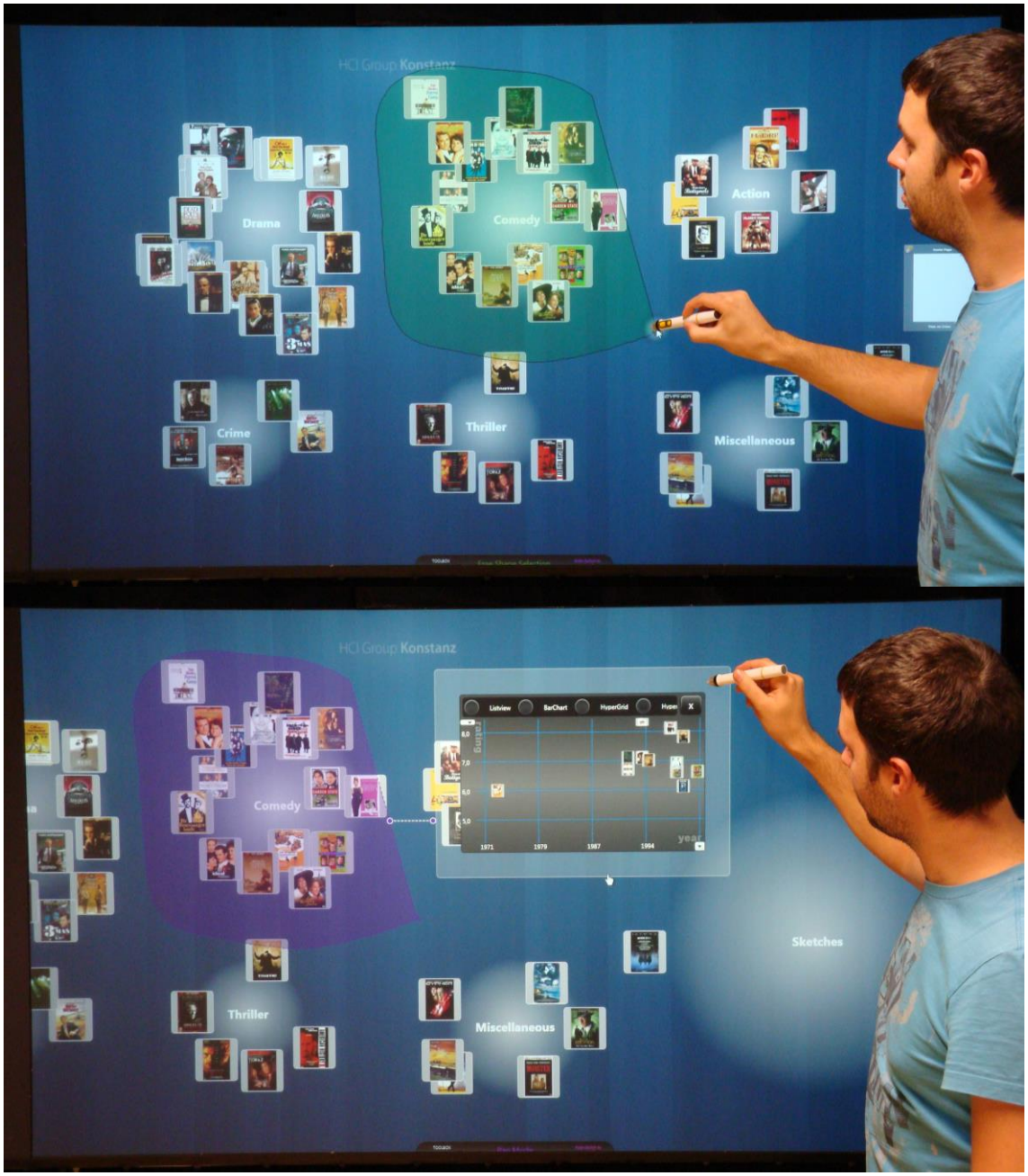


Figure 18 – Pen input on a vertical display can be used to draw a shape for selecting objects (top) and displaying a visualization of the contained objects (bottom).

Figure 19 – Different see-through visualizations (list, scatter plot, bar charts, HyperGrid) that can be displayed inside the tangible lens from Figure 91.

Figure 20 – Adding virtual Post-It notes using the 'Post-It' lens, touch input, and digital pen and paper.

2.2 DeskPiles

The *DeskPiles* prototype was developed as a part of a cooperation project between the HCI Group of the University of Konstanz, the Integrated Systems team at Microsoft Research Cambridge (henceforth MSRC), and the NanoPhotonics Centre at the University of Cambridge (henceforth NP).

As a starting point, researchers from MSRC studied the work practices of the nano-scientists working at NP and how they use information artifacts (*Oleksik, Milic-Frayling, and Jones 2012*). Based on this user research and a co-creation workshop with researchers, the project considered different potential uses of interactive tabletops in the NP environment. The project partners decided for the creation of a tool for presenting, discussing, and annotating lab results during the frequent meetings between scientists and the lab leader in his office. Therefore a tabletop and vertical screens were integrated in the office environment of NP in Cambridge and a similar setup for development and testing was created in Konstanz.

Figure 21 – Tabletop in the NP office environment at Cambridge (left). Setup for development and testing purposes in Konstanz (right). Source: DeskPiles Technical Report.

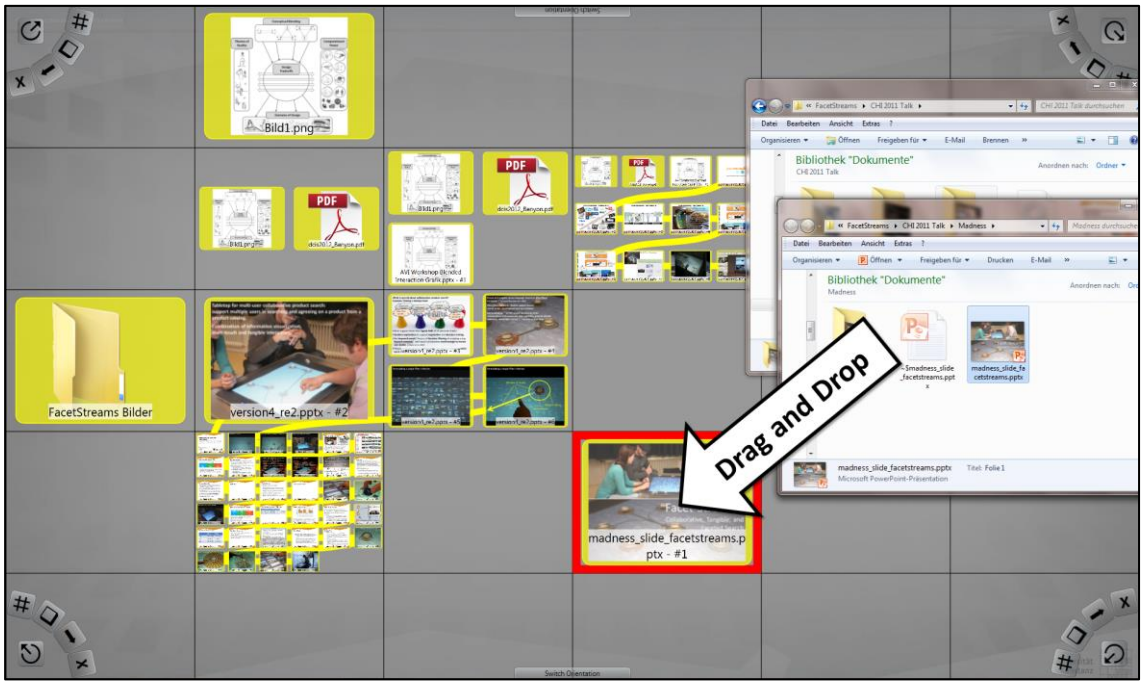


Figure 22 – In *DeskPiles*, the visual workspace is a grid structure that can be populated with information items from the file system.

The integration of tablet PCs into this setup was considered as particularly important, as all members of NP use tablet PCs with Microsoft OneNote as an electronic lab book. The tablets were used in lab and office environments to document the progress and results of experiments and lab work and to create summaries of their progress and results for biweekly meetings.

Because of our group's experience with designing table-centric interactive spaces and the availability of our ZOIL software framework, the HCI Group took the lead during the design and implementation of the *DeskPiles* prototype. For this purpose, I applied the

ZOIL design principles and used the ZOIL software framework together with Toni Schmidt and Michael Zöllner to design and implement a prototype application[13].

The *DeskPiles* application can be executed on all Windows-based devices such as the Microsoft Surface tabletop, tablet PCs, and desktop PCs. It processes Surface multi-touch input, stylus input, and mouse input and thus can be used on the different hardware platforms without device-specific alterations.

Unlike in the *Media Seminar Room*, the information landscape is structured by a grid that is populated by the users with information items from their file system such as images, slides from presentations, videos, or documents. The decision for a grid structure is based on experiences from the *Media Seminar Room*. Although multi-touch manipulations greatly facilitate the repositioning, resizing, and rotation of objects in the information landscape, users still found it difficult to establish regular spatial structures, e.g., to align several objects or to give them the same size. Therefore, *DeskPiles* employs mechanisms for the automatic alignment of objects. By dragging an information item in an empty cell, the item is scaled to the cell size. If further items are dragged into the cell, all items are automatically rescaled and repositioned, so that they appear in a regular grid layout (Figure 23).

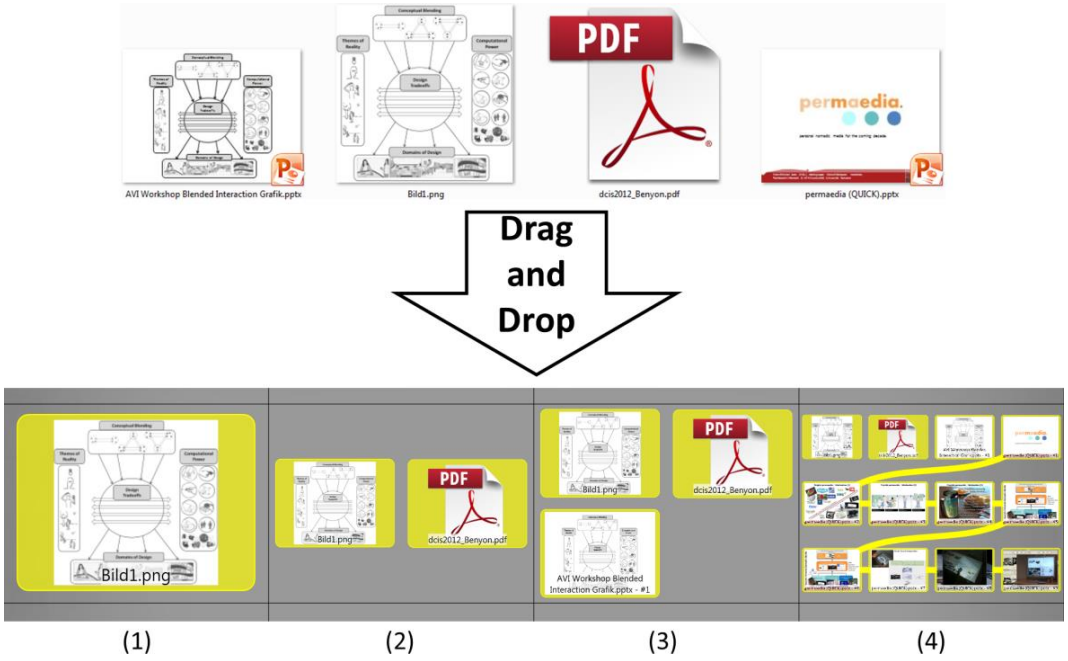


Figure 23 – Populating a cell with a growing number of objects in *DeskPiles*. From left to right: (1) single image object, (2) image & PDF, (3) image, PDF & single slide, (4) image, PDF, single slide & slide deck.

[13]A project description can be found at *http://research.microsoft.com/en-us/projects/deskpiles/* and a technical report about the project was published online at *http://research.microsoft.com/en-us/projects/beneath-the-surface/bts_cambridge_metasurfacing_with_the_surface_finalreport.pdf*. Furthermore, a presentation is available at *http://research.microsoft.com/en-us/um/redmond/events/ersymposium2010/slides/baumberg.pdf* and a video of the DeskPiles prototype can be found at *http://www.youtube.com/watch?v=yrz6PHXW7rM* (All accessed Jul 15, 2012).

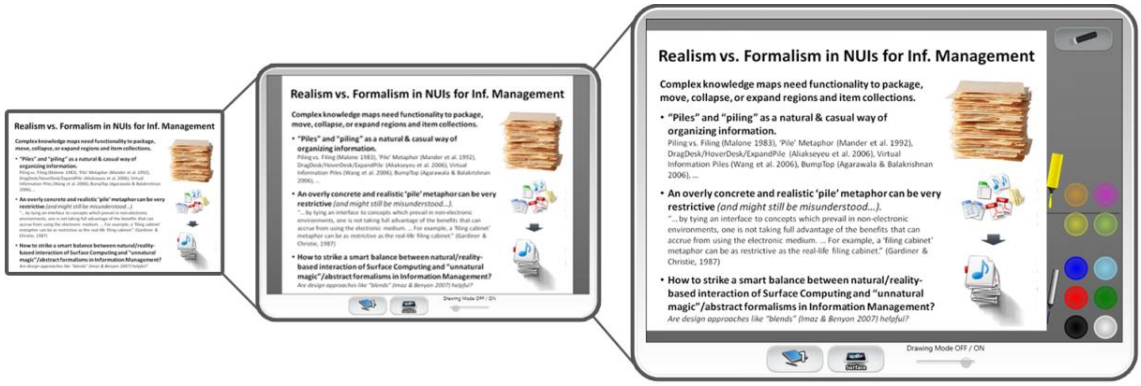


Figure 24 – Size-dependent representations with additional controls for view control and annotations.

Although the grid restricts the degrees of freedom that users have for establishing a spatial structure of items in the information landscape, the navigation can still be done by continuous zooming and panning and all items can be explored individually by semantic zooming. Similar to the *Media Seminar Room*, multiple representations of items are used to provide additional functionality (Figure 24), e.g., a toolbox for leaving marks and annotations on objects (Figure 25), or buttons to send the current view to a remote device, e.g., another tablet or the tabletop. Furthermore, users are provided with functionality to connect items with visual links to explicitly express a relationship and they can add free-floating virtual Post-It notes using handwriting or keyboard to label or comment items or regions. Figure 26 shows an example of an information landscape that was created during a user study with NP researchers.

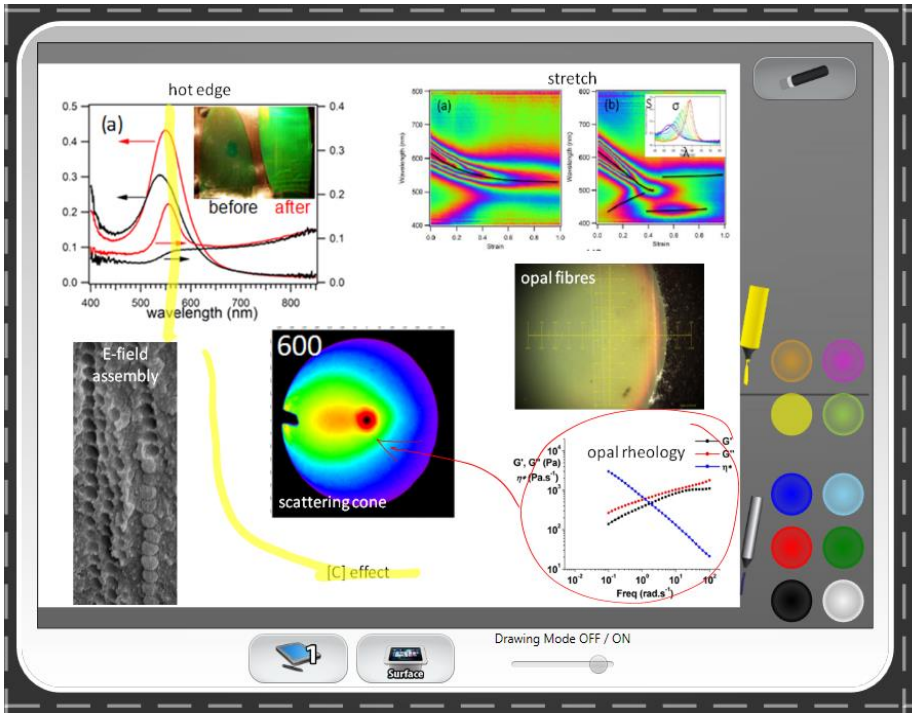


Figure 25 – *DeskPiles* enables users to annotate items with virtual ink after zooming in. On the tabletop, coarse grained annotations and highlights are made using fingers. For more fine grained annotations, users can use the stylus of connected tablet PCs.

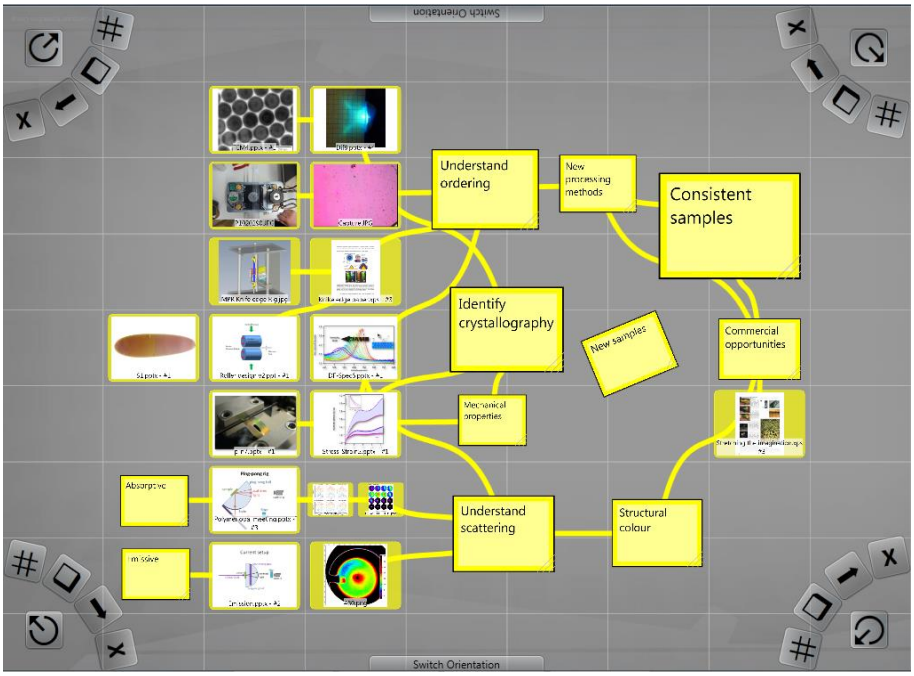


Figure 26 – An example of an information landscape created by a NP researcher during a user study.

A typical usage scenario of *DeskPiles* is described in the following narrative: Nick, a NP doctoral student and researcher prepares a summary of his lab work of the last two weeks for one of the biweekly meetings. The meeting will be held in the office of Peter, who is a professor and the leader of the NP lab. Furthermore, Jane will attend the meeting, who is a postdoctoral researcher and is working on a topic that is closely related to Nick's work.

During his work, Nick has encountered some surprising outliers in his data that could either hint at a relevant scientific finding or could be a result of an error in his experimental setup, e.g., faulty wiring or missing grounding of a high-precision measuring device. Nick hopes that the experienced researchers Jane and Peter can advise him on how he should proceed to avoid "hunting a phantom" and wasting important time and lab resources.

Nick starts his preparation by using the *DeskPiles* application on his tablet PC in his office to populate the grid with a selection of relevant information items that he has collected in his file system during lab work. For example, he has created a Powerpoint presentation containing screenshots from an application for analyzing and visualizing spectrographic data. Nick has also made some pictures of his current experimental setup with his mobile phone that he also imports into *DeskPiles*. Furthermore, he imports some pages from a PDF document and another Powerpoint presentation that describe the research goal and approach of Nick's project, so that he can give a very brief introduction at the beginning of the meeting, in case that not all participants are familiar with his work. To semantically structure the different information items, Nick puts closely related items into the same grid cell and uses proximity, visual links, or Post-It notes to establish a meaningful spatial structure (similar to Figure 26). Furthermore, he

prepares two empty items at the edges of the grid that he wants to use for personal notetaking during the meeting.

At the beginning of the meeting, Nick, Jane, and Michael connect their *DeskPiles* applications on their tablet PCs to a meeting server. Similarly, Peter connects the *DeskPiles* application on the tabletop to this server. By doing this, Nick's grid and the contained information items become visible on all connected devices and each participant can navigate the grid structure and browse its items individually (Figure 27).

Nick guides the other participants through his prepared materials on the tabletop and explains what he has done and experienced during lab work. On his tablet PC, he zooms on one of the empty items he has prepared to take personal notes using a stylus. During his presentation on the tabletop, one of his screenshots with a spectrogram catches the attention of Peter and Jane. For this reason, Nick uses the view control button on the tabletop to send this view to Jane's tablet PC. Peter and Jane discuss the spectrogram that is now visible on the tabletop and on the tablet PC (Figure 28).

The spectrogram reminds Jane of a problem that she has encountered in a similar experiment. Therefore she uses the high-resolution and pressure-sensitive stylus of her tablet PC to mark and label interesting bits of the spectrogram, so that Nick gets feedback on what he should pay attention to. These annotations appear immediately on all other connected devices including the tabletop. Since neither Jane nor Peter are sure, if this is a relevant finding or a problem with the experimental setup, Peter suggests to use an additional filtering device, so that Nick can verify his measurement with an improved setup. To collect suggestions for the next steps, Peter moves to a pen-enabled vertical display that also runs a *DeskPiles* application that is connected to the meeting server (Figure 29). The participants discuss further TODOs and Peter writes them into the second empty item that Nick has prepared for notetaking to conclude the meeting.

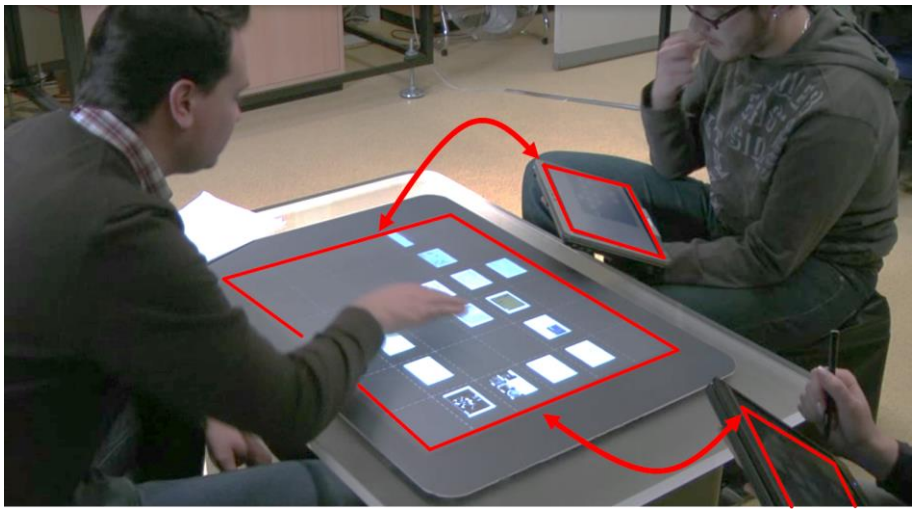


Figure 27 – *DeskPiles* running on a tabletop and two tablet PCs. They are connected to the same meeting server to provide a shared information landscape on all devices.

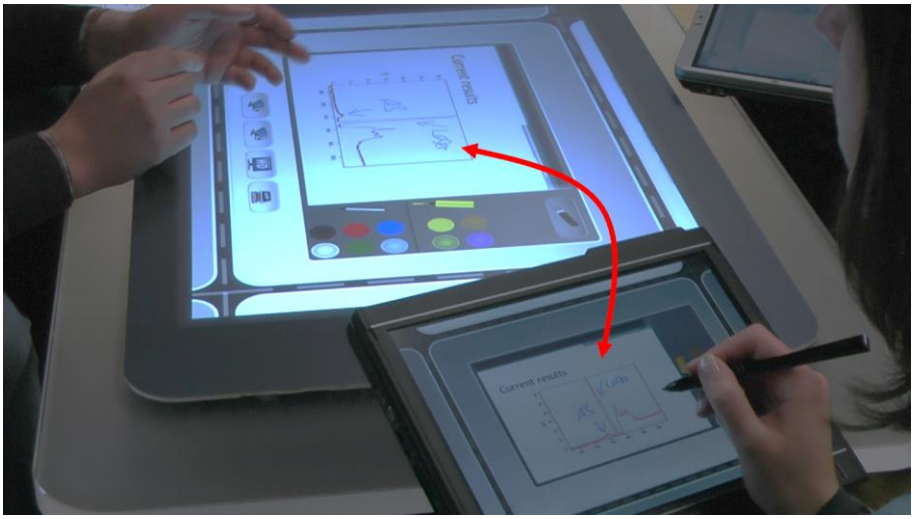


Figure 28 – For in-depth discussion and annotation, a visual information item is displayed simultaneously on the tablet PC and tabletop that are connected via the meeting server.

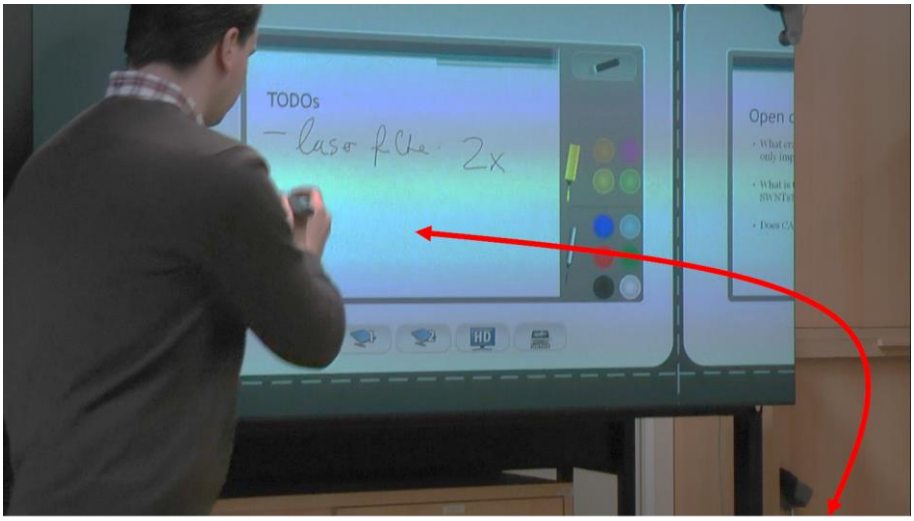


Figure 29 – To conclude the meeting, a participant collects a list of TODOs on a vertical pen-enabled display that is also connected to the meeting server.

2.3 Distributed Sketching

The *Distributed Sketching* prototype explores the use of ZOIL-based interactive spaces for groupwork in creative design (*Geyer, Jetter, Pfeil et al. 2010*). Figure 30 shows pictures from the resulting prototype[14].

Following the ZOIL paradigm, the prototype is implemented using the ZOIL software framework and uses ZOIL's zoomable information landscape to mimic a pin board that contains and organizes sketches in space and scale (Figure 31). This spatial scheme of organizing artifacts is inspired by the observations of Vyas et al. about the important role of space as a resource in creative design practices and in present-day design studios

[14] A video of an early prototype is available at *http://hci.uni-konstanz.de/downloads/collaborative_sketching.wmv* (Accessed Jul 15, 2012)

or similar work environments (*Vyas, Veer, Heylen et al. 2009*). Furthermore, the prototype also enables to add image objects or slides to the information landscape using drag and drop from the file system. They can serve as *"informative, inspirational and creative design-related artifacts"* (*Vyas, Veer, Heylen et al. 2009*).

Figure 30 – *Distributed Sketching* in a ZOIL-based interactive space. Top: Early version of the prototype from (*Geyer, Jetter, Pfeil et al. 2010*). Bottom: More recent version from (*Reiterer 2011*).

Figure 31 – In the *Distributed Sketching* prototype, ZOIL's information landscape mimics a large zoomable pin board for spatially organizing sketches and inspirational artifacts for creative design.

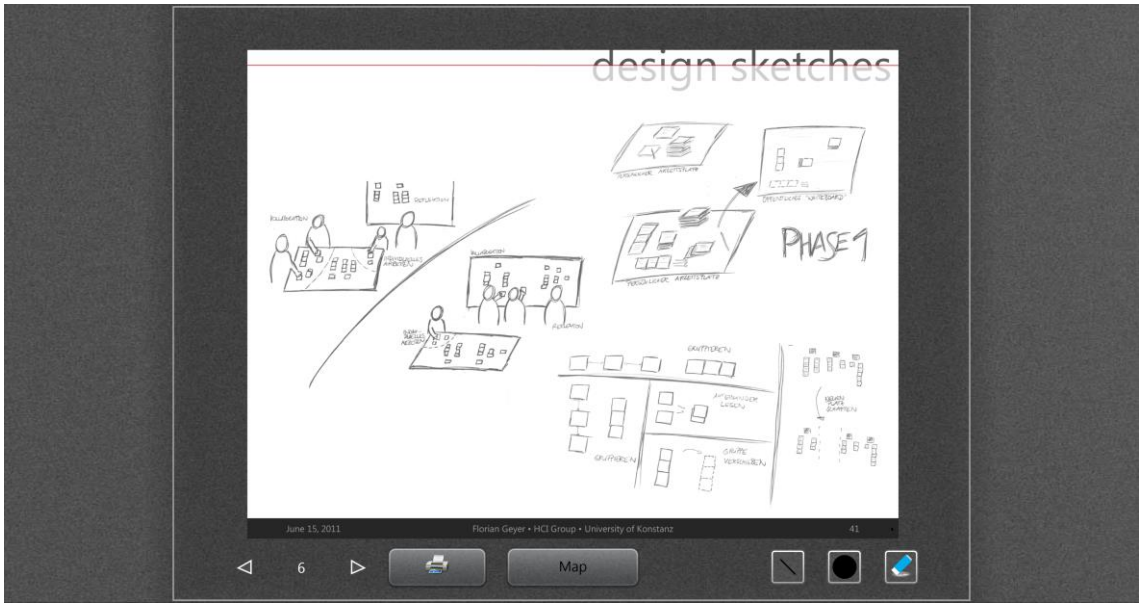


Figure 32 – The content of sketches or artifacts can be revealed by zooming and panning.

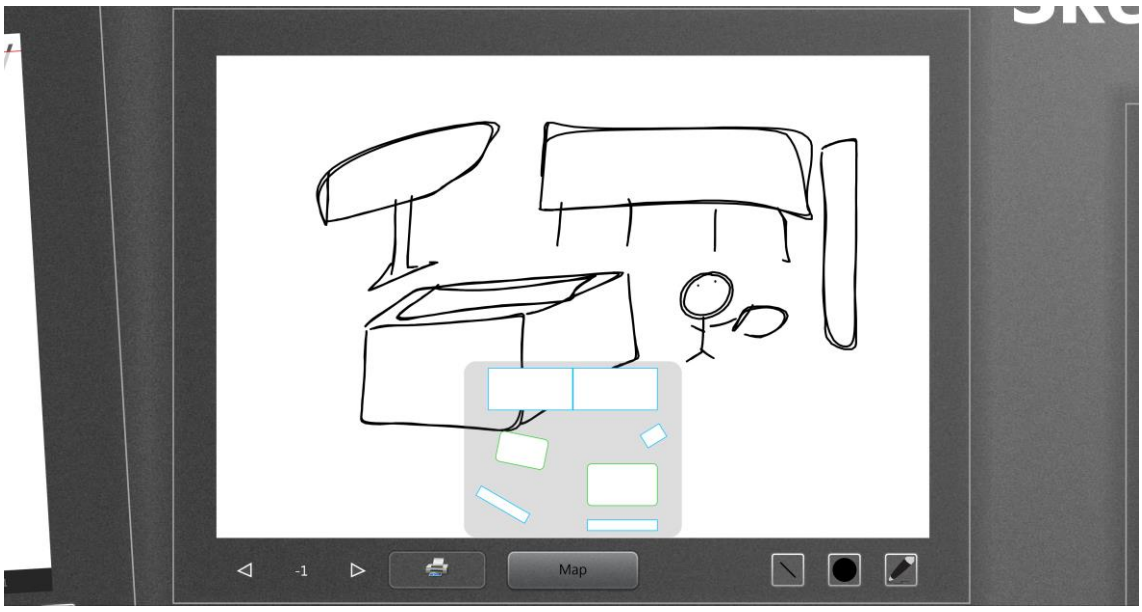


Figure 33 – The "Map" button opens a map with the different devices in the interactive space. The current view can be sent to remote devices by selecting their representation in the map.

The pin board can be accessed from all devices. Tasks like sorting or discussing sketches are supported in the same way that movie objects can be accessed and sorted in the *Media Seminar Room*. Furthermore, Anoto digital pen & paper technology is used to provide pen-and-paper only interaction. New sketches can be created by simply grabbing a blank sheet of paper and starting to draw, while existing sketches can be printed out for annotation or refinement (see "Printer" button in Figure 32).

By supporting multiple pens and devices at the same time, different styles of tightly-coupled and loosely-coupled collaboration are supported, e.g., distributed reviews or reflection. To facilitate this, the prototype provides a "world-in-miniature" technique for

assigning views to different devices (*Nacenta, Gutwin, Aliakseyeu et al. 2009*): By pushing the map button, a map with the different devices in the interactive space from a bird's eye perspective appears and enables users to flexibly send the current view to other remote devices by touching their representation in the map (Figure 33).

2.4 Facet-Streams

As described in the introduction of this thesis, an important part of collaborative knowledge work is searching and acquiring information from databases. For this reason, the *Facet-Streams* prototype specifically explores how multiple users can collaboratively search for information using faceted navigation in a catalog of products or similar information items based on the items' metadata (*Elmqvist, Vande Moere, Jetter et al. 2011*; *Jetter, Gerken, Zöllner et al. 2010a*; *Jetter, Gerken, Zöllner et al. 2011*).

Since *Facet-Streams* is focused on collaborative search around a tabletop (Figure 34), it cannot be considered an interactive space but focuses specifically on the design challenge of multi-user around-the-table interaction. However, it thereby demonstrates how a tabletop can be used to integrate faceted search functionality into an interactive space[15]. Figure 35 illustrates a more recent design of *Facet-Streams* currently under development that is integrated in a multi-device environment with multiple displays and pen input. Nevertheless, the focus of *Facet-Streams* in this thesis is only on the interaction design for a single tabletop multi-user system.

Facet-Streams supports small groups during decision-making and negotiation by enabling a faceted exploration of a product catalog, e.g. a catalog of hotels for a family's vacation. The content of the catalog is presented on the tabletop as a grid structure that can be explored by zooming and panning using multi-touch manipulations. Zooming into a hotel object reveals hotel data of different facets, e.g., name, location, stars, price per night, and also user-generated content such as photos of hotel rooms (Figure 36).

Figure 34 – The *Facet-Streams* tabletop system for collaborative faceted search. Source: (*Jetter, Gerken, Zöllner et al. 2011*).

[15] Video of Facet-Streams at *http://www.youtube.com/watch?v=giDF9lKhCLc* (Accessed Jul 20, 2012).

Figure 35 – In future, *Facet-Streams* will be integrated as a search tool in a multi-device environment with vertical screens and pen input (see PhD project of Roman Rädle, HCI Group, Konstanz).

Queries can be formulated by each participant by putting small glass discs as tangible facet tokens on the tabletop. By interacting with the virtual controls around them, users' can select the desired facet and the desired value ranges (Figure 37). The facets and value ranges of a node are changed by touching and sliding the finger over the token's facet and value wheel controls. This also enables users to develop more advanced techniques, e.g. bi-manual selection of value ranges during which one hand rotates the glass token and its attached wheel while a finger of the other hand selects the segments of the wheel that are rotating below.

These tokens can then be visually linked to form a directed graph that serves as a visual filter/flow representation (*Young and Shneiderman 1993*) of faceted Boolean search. All products from the catalog flow along the edges of the network and are filtered by the nodes they pass. Logical AND and OR can be expressed without mathematical notations or query languages, simply by connecting nodes or letting edges flow together. Query results can be inspected by the users by touching an edge to reveal the products that flow therein. Alternatively, a different result token can be put permanently on any stream within the network to reveal which hotels are contained there. The subset of contained hotels is visualized by only showing these hotels in the zoomable grid in the background of the visual-tangible filter/flow representation.

Figure 36 – The grid structure on the tabletop with the content of the catalog. Details and user-generated content can be accessed by zooming in. Source: (*Jetter, Gerken, Zöllner et al. 2011*).

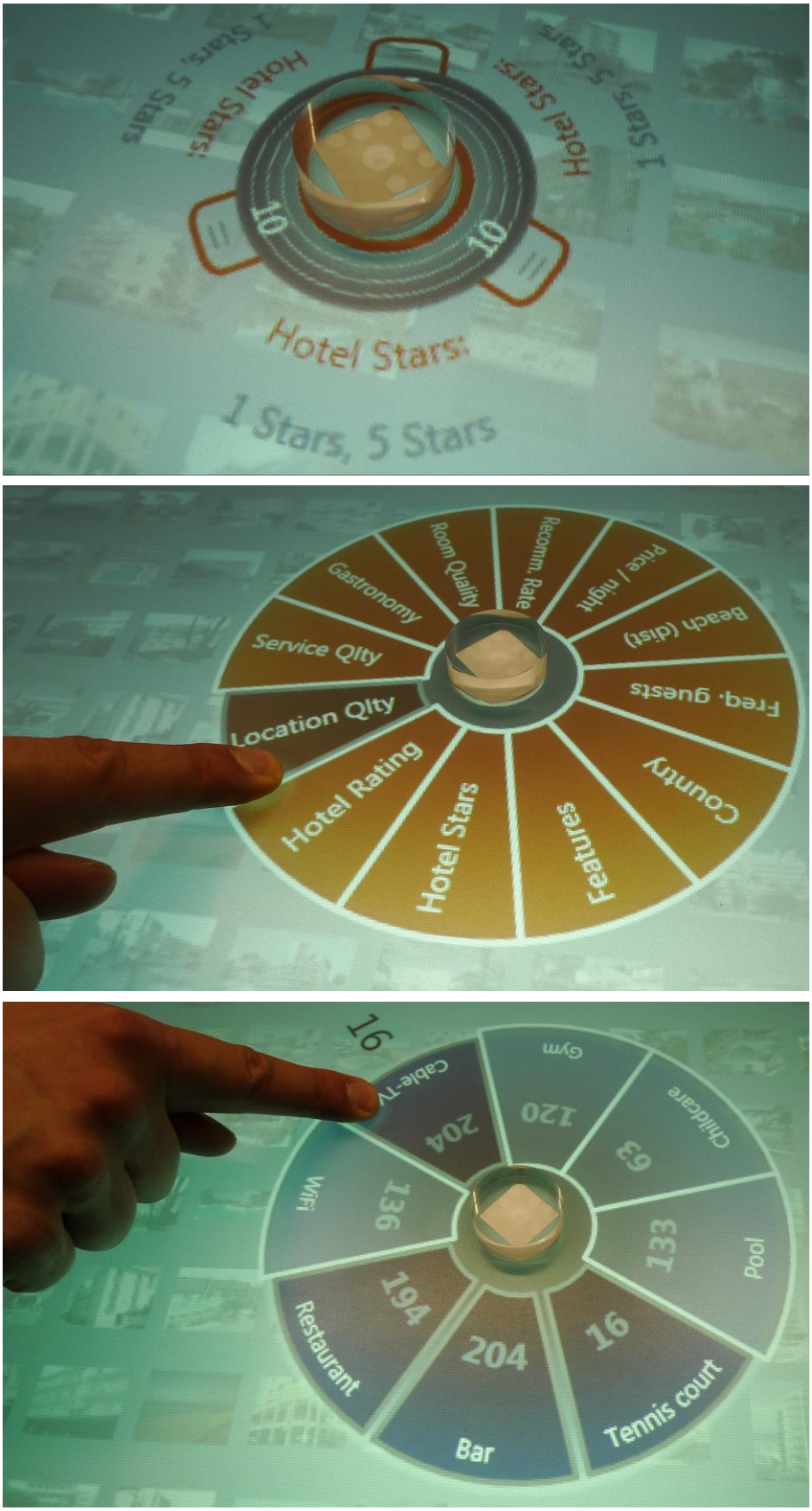


Figure 37 – A facet token can be used to select a data field as a facet by sliding the finger over the facet wheel (e.g., 'location quality', 'hotel features'). The desired values for this facet can be selected by tapping or sliding over the value wheel (e.g., 'Cable TV', 'Restaurant').
Source: (*Jetter, Gerken, Zöllner et al. 2011*).

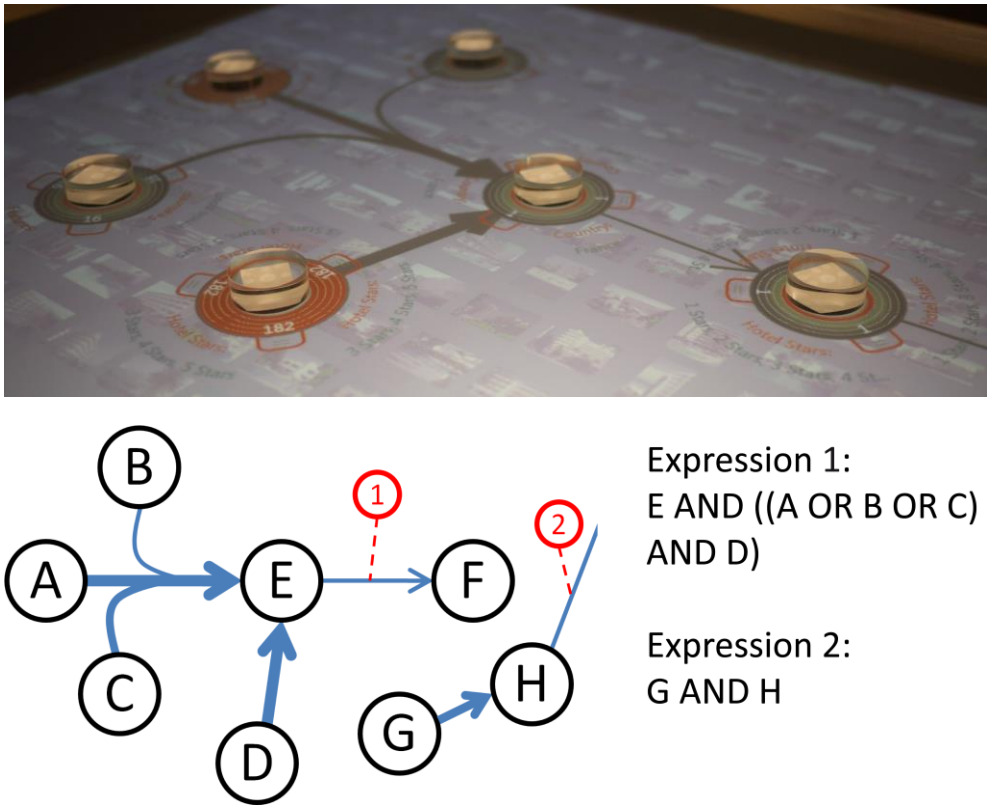


Figure 38 – Facet tokens can be connected to a filter/flow representation of faceted Boolean search to formulate complex queries (top). The network of tokens with their contained criterion are internally translated into Boolean expressions for querying the database (bottom).

Through tangible and touch input, *Facet-Streams* exploits a greater range of the users' real-world motor skills than non-tangible and non-multi-touch interfaces. The number and spatial layout of nodes can be easily altered by familiar physical manipulations, similar to placing, lifting or, moving the pieces of a board game like checkers. The topology of the network can be changed by touch interaction, e.g. by dragging new connections between nodes with the fingers or by cutting them with a crossing out gesture. Thereby every kind of tangible and touch input into the system leads to immediate visual feedback. For the users, this creates the illusion of direct physical interaction with the visual representation.

The benefit of this design is a low viscosity of the query's visual representation, i.e., a *"low resistance to change"* in the interface (*Jetter, Gerken, Zöllner et al. 2011*). It enables users to rapidly modify it according to their individual or shared goals. During initial search phases, each group member can formulate and explore their own criteria individually during phases of loosely-coupled parallel work. These personal query networks can then be effortlessly combined into a larger group network for collective reviewing during phases of tightly-coupled collaboration. However, the query's network can easily be dissolved into smaller parts again, e.g., for returning to parallel work, or to separate the satisfactory parts from those that need further refinement. Therefore, low viscosity also gives the necessary flexibility to support different working styles or different collaborative phases.

2.5 Summary

To summarize the characteristic interaction and visualization techniques of the presented ZOIL-based interactive spaces, Figure 39 gives an overview of the employed ZOIL design principles and in which of the following chapters the principle is described and discussed in detail.

	Media Seminar Room	***DeskPiles***	***Distributed Sketching***	***Facet-Streams***
P#1: Provide post-WIMP functionality as objects, not applications (chapter 3)**.**	No applications. Direct manipulation of objects. Functions such as playing video or writing on virtual Post-It notes are directly attached to the information items.	No applications. Direct manipulation of objects. Polymorphism: All objects (e.g., slides, images, documents) behave similar regardless of their kind.	No applications. Direct manipulation of objects. Polymorphism: All objects (e.g., slides, images, sketches) behave similar regardless of their kind.	Formulation of search criteria by direct manipulation of tangible facet tokens and virtual links. Object-action style of interaction with multi-touch controls around tangible tokens.
P#2: Provide a zoomable user interface for navigation with semantic zooming (chapter 4)**.**	Zooming & panning in information landscape with movie objects and Post-It notes. Semantic zooming for accessing details and video playback.	Zooming & panning in information landscape with scientific information items. Semantic zooming for accessing details and drawing functions.	Zooming & panning in information landscape with design artifacts. Semantic zooming for accessing details and drawing functions.	Zooming & panning in information landscape with hotels. Semantic zooming for accessing details and user generated content.
P#3: Provide space for sensemaking, marks, and annotations. (chapter 5)**.**	Objects can be freely arranged and clustered in space and scale. Virtual Post-It notes can be used for annotations with digital pen & paper.	Objects can be arranged in grid structure. Virtual Post-It notes can be used for annotations. Toolbox for drawing on items. Possibility to create visual links.	Objects can be freely arranged and clustered in space and scale. Post-It notes can be used for annotations with digital pen & paper. Toolbox for drawing on items.	Tangible facet tokens can be freely arranged in physical space on the tabletop to create spatial representations of the search process..
P#4: Provide space for coordinating mixed-focus collaboration (chapter 5)**.**	Individual “cameras” to provide personal or shared views of the zoomable information landscape to enable ad hoc partitioning of workspace.	Individual “cameras” to provide personal views of the zoomable information landscape to enable stylus input on tablets.	Individual “cameras” to provide personal or shared views of the zoomable information landscape to enable ad hoc partitioning of workspace..	Tangible facet tokens can be freely arranged and connected for fluid transitions between individual and group work.
P#5: Provide post-WIMP InfoVis tools for fluid interaction (chapter 6)**.**	InfoVis tools for movie meta-data using visualization lens or by drawing shapes around clusters.	N/A	N/A	Faceted navigation in hotel meta-data using tangible facet tokens connected by an InfoVis Filter/Flow metaphor.
P#6: Support multi-user collaboration with visual-tangible externalizations (chapter 6)**.**	N/A	N/A	N/A	Visual-tangible externalization of the search process using a tangible Filter/Flow metaphor.

Figure 39 – Summary of the presented prototypes and their characteristic interaction and visualization techniques and their relation to the ZOIL design principles in the following chapters.

3 Post-WIMP Object-Oriented User Interfaces

This chapter introduces the concept of *Object-Oriented User Interfaces* (OOUIs) and applies it on post-WIMP interaction in interactive spaces. Within ZOIL, *object-oriented* instead of *application-oriented* interaction is important for achieving a seamless interaction with different information types and tools. In particular, OOUIs can help to overcome the boundaries between WIMP application software with its many different competing interaction styles, conceptual models, data formats, and storage locations.

The first part of this chapter is about understanding and designing OOUIs. It revisits and summarizes different views of OOUIs in HCI literature of the WIMP era and enters new terrain by applying them on post-WIMP collaborative knowledge work. Here, the scientific contribution lies in the theoretical discussion of those properties of OOUIs that are particularly helpful with regard to ill-defined and strongly *situated* and *embodied interactions* during collaboration in interactive spaces. Furthermore, the first part also discusses the use of object-oriented design principles such as *inheritance* and *polymorphism* to achieve greater consistency and usability of post-WIMP UIs. The first part concludes with the results of a small case study with student designers and developers about the applicability of OOUI design and modeling techniques in the context of ZOIL and the formulation of the *1st ZOIL design principle*.

The second part of this chapter is about implementing OOUIs for post-WIMP interactive spaces in multi-user, multi-display, and multi-device settings. For this purpose, it introduces the *ZOIL software framework* that serves as a kind of middleware between the application level and the operating system and provides high-level functionality in the areas of presentation & interaction, network communication, and persistence & synchronization. To achieve this, it uses the software design patterns of *Transparent Persistence*, *Model-View-ViewModel,* and *Attached Behaviors*. The second part concludes with two evaluation studies of the ZOIL software framework's API usability with student designers and developers and a discussion of the framework's practical value.

This chapter includes parts of a full paper on ZOIL and OOUIs that was published at HCSE 2010 (*Jetter, Gerken, Zöllner et al. 2010b*). It also contains the parts about OOUIs and their role in ZOIL from a workshop paper at CHI 2008 (*Jetter, König, Gerken et al. 2008*) and two book chapters about the ZOIL software framework (*Jetter, Zöllner, and Reiterer 2011*; *Zöllner, Jetter, and Reiterer 2011*). This chapter uses the empirical methodology of concept maps for API usability evaluation introduced in a CHI 2011 full paper (*Gerken, Jetter, Zöllner et al. 2011*) and the description of ZOIL's software architecture and the evaluation results from a journal article on the ZOIL paradigm (*Jetter, Zöllner, Gerken et al. 2012*).

3.1 Introducing Object-Oriented User Interfaces

The underlying assumption of *Object-Oriented User Interfaces* (OOUIs) is that UI and interaction design can benefit from the same principles that made *Object-Oriented Programming* (OOP) and programming languages such as C++, Java or C# so remarkably successful. Like OOP, OOUIs are based on the hypothesis that *object-orientation*, i.e., understanding, analyzing, and modeling the world in terms of *objects* and their *classes*, is a natural and very efficient way of conceptualizing our environment and is close to the way we reason in our everyday world (*Collins 1995: 77*; *Søgaard 2006*). This is assumed for UI designers and developers, but likewise for the users of a user interface.

By conceptualizing a real-world domain (e.g., a public library, a bakery, a game of football, or a brainstorming session) in terms of its relevant objects, their classes, and mutual relations, designers and developers can construct an object-oriented *simulation model* of the domain (*Collins 1995: 70*). To achieve this, all relevant classes of domain objects (e.g., book, lender, baker, player, ball, card), their *relations* (e.g., is a, has a, owns a), their *attributes* (e.g., name, position, color), and attached *functions* or *behaviors* (e.g., create, add, remove, give) become part of this simulation model[16]. This analysis and modeling of a specific problem domain from the real world in terms of objects, classes, attributes, functions, behaviors, and relations is at the heart of object-orientation and is the *"meta-model"* (*Collins 1995: 70*) that OOP and OOUIs share.

According to OOUI proponents, this object-oriented analysis and modeling of a domain does not only help programmers to implement the necessary business logic and data models in an object-oriented programming language, but it also helps interaction designers to systematically design the user interface. An OOUI designer uses the simulation model to create an UI that presents the application domain to the end-user as a simulated world of cooperating objects with selected attributes, behaviors, functions, and relations taken from the real world. The simulation model becomes visible on the screen and users can use direct manipulation to directly interact with all of its objects to complete their tasks. There is a concrete mapping between the domain's real-world entities or concepts and the UI objects and their interactive behavior. Users familiar with the domain can easily relate to the UI's objects and learn how to use them. *"Users see and manipulate object representations of their information. Each different kind of objects supports actions appropriate for the information it represents. Users of object-oriented computer user interfaces need not to be aware of computer programs and underlying computer technology in order to use computers"* (*Roberts, Berry, Isensee et al. 1998: 11*).

As a consequence of this direct and concrete mapping of domain to UI objects, an OOUI *"focuses the users on objects – the 'things' people use to accomplish their work"* (*Roberts, Berry, Isensee et al. 1998: 11*) instead of primarily focusing on functions or processes and making sequential work procedures the first-level citizen of the UI. This focus on objects

[16] Not surprisingly, one of the earliest object-oriented programming languages SIMULA was created for the purpose of simulating real-world physical processes (see Ole-Johan Dahl and Kristen Nygaard, 'Simula: An Algol-Based Simulation Language', *Commun. ACM*, 9/9 (1966), 671-78.).

is supposed to be closer to the way that users naturally conceptualize their physical and social work environment. Instead of following given work procedures or processes, users can act more naturally and flexibly in their virtual work environment that provides familiar objects and the necessary tools to perform their tasks. Ideally, this leads to *"expressive"* UIs (*Pawson, Bravard, and Cameron 1995*). Such expressive UIs focus on the user's objects and they do not impose predefined processes or sequences of actions. They do not treat the user *"(...) merely as a dumb process-follower (...)", but as an "(...) empowered problem-solver"* (*Haywood 2009: xii*). Ideally, they leverage users' creativity and problem-solving skills and feel more like using a drawing program than a business system (*Haywood 2009: xii*).

The notion of an object-oriented UI was popularized as a core principle of IBM's OS/2 operating system in the 1990s. In OS/2, the traditional packaging of functionality in applications was meant to be replaced by a set of cooperating objects and their views which are integrated into a unified graphical work environment under a consistent interaction model (*Mandel 1994*). However, except for IBM's Workplace Shell for the OS/2 operating system (*Berry and Reeves 1992*) and IBM's OVID methodology (*Roberts, Berry, Isensee et al. 1998*), strictly object-oriented UIs have not become widespread in HCI practice or research. Only selected features of OOUIs such as drag-and-drop, WYSIWYG[17], and pop-up context menus with an "object-action" or "noun-verb" style of interaction have found their way into popular GUIs such as Microsoft Windows and its successors. The surrounding WIMP desktop metaphor remained strongly application-oriented with many competing application metaphors and interaction styles, e.g., point & click of textual commands and hyperlinks in page-oriented Web applications versus direct manipulation of visual icons and content in WYSIWYG editors. Even recent versions of the desktop metaphor are still based on isolated applications that reside in distinct application windows and use proprietary file formats. Therefore they are criticized for not living up to the flexibility, powerfulness, and seamlessly integrated working procedures that are claimed to have become reality by now (*Ravasio and Tscherter 2007*).

Although HCI researchers have repeatedly criticized the traditional application-oriented desktop metaphor and also suggested alternative interaction models, e.g., *instruments* (*Beaudouin-Lafon 2000*) or *commands* (*Raskin 2000*), I am not aware of any work in HCI that has objectively analyzed the advantages and disadvantages of OOUIs or discussed why they failed on the market. For example, Mandel presents results of a usability test of the OS/2 Workplace Shell with very positive results for OOUIs (*Mandel 1994: 305*), but the study's design and analysis do not live up to the standards of today's scientific HCI research. Unlike Mandel, Constantine and Lockwood harshly criticize the idea of object-oriented interaction: *"Only a programmer whose mind has been warped by too many years of small talk with object-oriented programming systems would conceive of interaction in this way"* (*Constantine and Lockwood 1999: 469*). However, their criticism is

[17] WYSIWYG = "what you see is what you get". Implies that an application shows a realistic depiction of its domain, e.g., a word processor enables user to edit a realistic preview of a document to be printed.

not based on any user studies or other empirical data. Furthermore, it is targeted towards the rigorous message-based object-orientation in the Smalltalk programming language that is described by Collins only to illustrate different OO meta-models (*Collins 1995*). Constantine and Lockwood falsely assume that *"users would be forced to interact with an object-oriented interface by moving little messages around from object to object on the screen"* (*Constantine and Lockwood 1999: 468*). In reality, OOUI interactions are based on direct manipulation and context menus and, although users create and manipulate objects, many users will never have to be familiar with OO principles or the underlying OO class hierarchy (*Roberts, Berry, Isensee et al. 1998: 16*).

A further explanation of the lack of research in OOUIs' is based on purely economic considerations: The great commercial success of the application-oriented Microsoft Windows operating system drove OS/2 out of the consumer market, because Windows was bundled with most new computers and supported a great variety of hardware. OS/2, however, was an expensive stand-alone software package that always suffered from a lack of hardware drivers. Possibly, OOUIs disappeared together with OS/2 from the consumer market and thus also from the focus of HCI research. A similar explanation could be the great success of the World Wide Web with its page-wise point & click Hypertext interaction in the late 1990s. This kind of interaction is opposed to the direct manipulation of graphical objects in OOUIs and, until a few years ago, such interaction was not available reliably on the Web. However, all these explanations must be considered speculative.

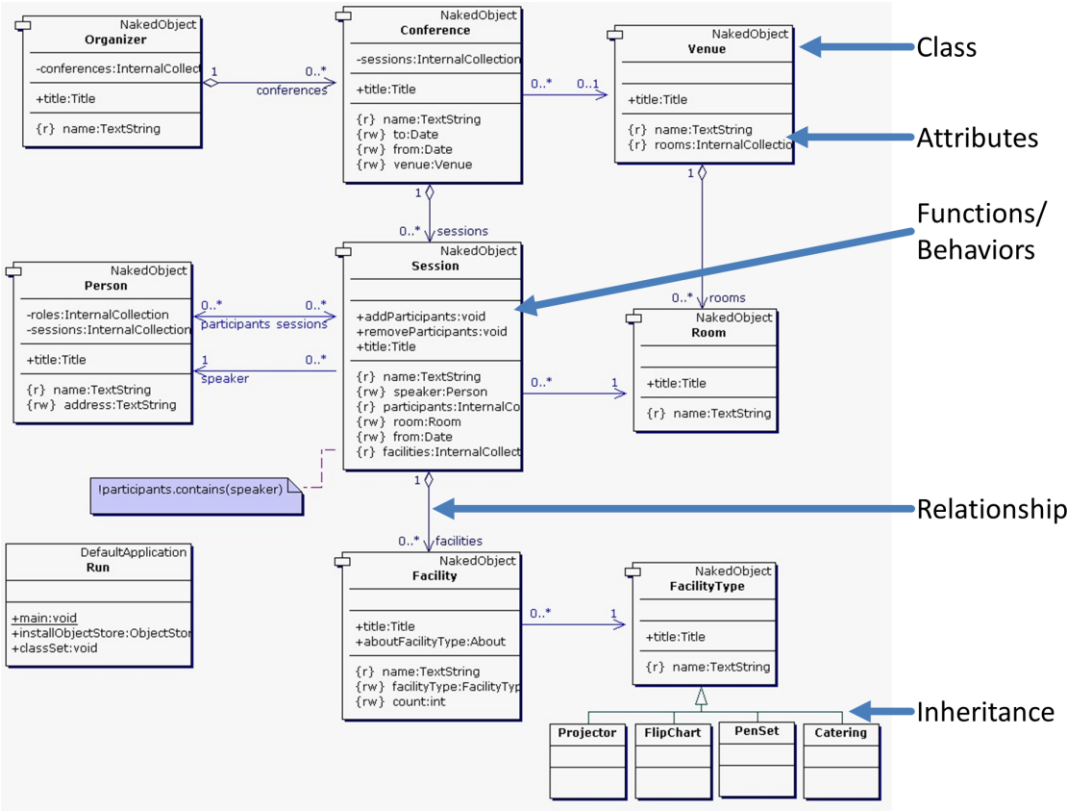


Figure 40 – An example simulation model for a conference management system in a UML-like notation. Adapted from: *http://nakedobjects.org/book/section44.html*

Despite the commercial failure of OOUIs, researchers and practitioners in Software Engineering have more recently rediscovered their interest in OOUIs. Examples for this are the 'Naked Objects Pattern' (*Pawson 2004*) and 'Domain-Driven Design' approaches (*Evans 2004*; *Haywood 2009*) for business systems. Their goal is to rapidly turn an UML[18] object model of the domain (Figure 40) into a working UI prototype (Figure 41) to engage business managers and involve end-users during design and development. Furthermore, this could *"deliver an expressive, object-oriented user interface for free"* (*Haywood 2009: xii*). However, the automatically generated single-user GUIs strongly rely on multiple windows and the traditional WIMP paradigm with an arguable usability and user experience (*Constantine 2002*).

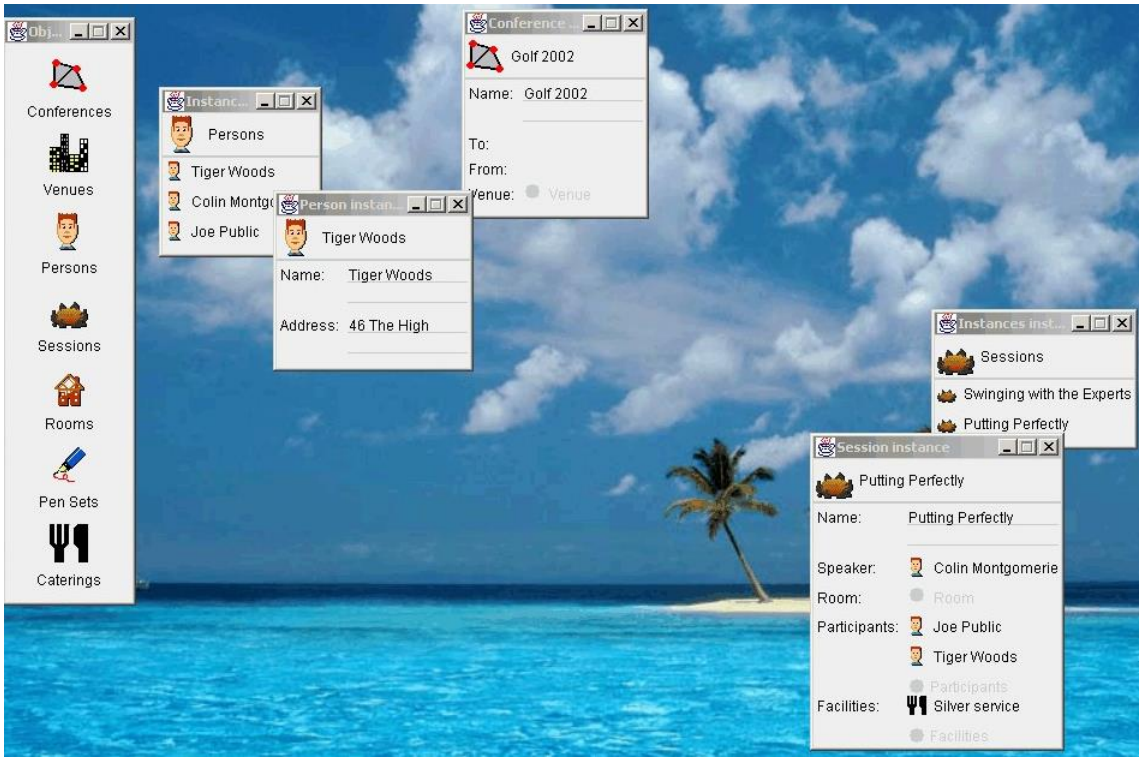


Figure 41 – The UI prototype generated from the model in Figure 40 using Naked Objects. Source: *http://nakedobjects.org/book/section44.html*

3.1.1 Operational Definition of OOUIs

When reviewing OOUI literature, it becomes obvious that although individual properties and characteristics of OOUIs are easy to describe there is no commonly accepted definition of OOUIs. For example, in their review of OOUIs, Constantine and Lockwood criticize Collins (*Collins 1995*) for defining OOUIs by what users are doing and what a good user interface is instead of actually defining OOUIs (*Constantine and Lockwood 1999: 468*). For the purpose of this thesis and to clarify the notion of OOUIs that I refer to, I do not intend to provide this missing definition, but I summarize the defining

[18] The Unified Model Language (UML) by the Object Management Group (http://www.omg.org) is the industry standard for modeling languages in the field of object-oriented software engineering.

characteristics of OOUIs that are discussed in HCI and Software Engineering literature and that are relevant for designing ZOIL-based interactive spaces as follows:

- An OOUI focuses the users on objects – the "things" people use to accomplish their work (*Roberts, Berry, Isensee et al. 1998: 11*). Domain and task analysis provide the basis for identifying the necessary user objects (*Roberts, Berry, Isensee et al. 1998: 14*). Users achieve their goals by gradually massaging these objects (*Nielsen 1993: 59*).
- Users see objects instead of applications (*Mandel 1994: 241*). There don't seem to be identifiable applications in an OOUI; there is merely a constellation of objects that a user can consider and manipulate (*Tibbetts 1991*). This resonates with present-day design principles like *"the content is the interface"* for natural user interfaces (*Hofmeester and Wixon 2010*) that recommend to focus UIs on their content and to use only a minimum of administrative controls[19] or system states (*Wigdor and Wixon 2011: 31*).
- Interaction with OOUIs is based on direct manipulation and users get immediate feedback from actions. The interface is modeless and displays objects in WYSIWYG form. If commands are necessary, the syntax of commands is "object-action" (*Tesler 1983*).
- In any case, the distinctions between objects of an OOUI must be clear and useful, and the interface must be consistent (*Roberts, Berry, Isensee et al. 1998: 27*) to reduce the number of commands or operations the user needs to learn and remember (*Collins 1995: 91*). To achieve this, an OOUI defines an object-oriented class hierarchy for user objects (*Roberts, Berry, Isensee et al. 1998: 15*). This hierarchy lets users classify objects based on how they behave. In the context of what users are trying to accomplish, all the interface objects fit together into a coherent overall representation (*Collins 1995: 89*). When designers carefully define and clearly distinguish object classes and their behavior in the class hierarchy, they enable users to learn and predict the behavior of objects more easily (*Roberts, Berry, Isensee et al. 1998: 16*). However, although users create and manipulate objects, many users will never have to be consciously aware of which class an object belongs to, because a good hierarchy reflects users' existing knowledge, expectations, and experiences (*Roberts, Berry, Isensee et al. 1998: 16*).
- An OOUI is expressive and treats the user as an empowered problem solver (*Pawson 2004*). There is a flexible structure-by object. Users may perform task in their own way or innovate (*Mandel 1994*).
- Real-world metaphors are helpful for shaping the user's mental model of an OOUI (*Roberts, Berry, Isensee et al. 1998: 31*). However, blindfold realism can be disruptive.

[19] Information designer Edward Tufte formulated such a principle in his design critique of the Apple iPhone in 2008: *"The idea is that the content is the interface, the information is the interface – not computer administrative debris"* (*http://www.edwardtufte.com*) (Accessed Nov 5, 2012).

An OOUI does not simply use a high-fidelity simulation of the real-world as user interface, but focuses only on task-relevant objects, behaviors, attributes, and relations.

The following sections elaborate on these different characteristics and explain why ZOIL uses OOUI approaches for post-WIMP interactive spaces to overcome the limitations of traditional WIMP applications and the desktop metaphor.

3.1.2 OOUIs and Applications

In an OOUI, it is not the applications, but the objects of the application domain (e.g., documents, notes, sketches, images, videos) that are the first-level citizens of the user interface. *"Users see objects instead of applications"* (*Mandel 1994: 241*). The necessary functions to work with these objects are directly attached to them. Functions are not scattered over multiple walled applications or other silos of functionality. Instead, the user has complete control by directly manipulating the objects without a diversion to application windows or menus. *"Note that there don't seem to be identifiable applications in an OOUI; there is merely a constellation of objects that a user can consider and manipulate. In a very real sense, the user is the application"* (*Tibbetts 1991*).

Nielsen writes *"Object-oriented interfaces are sometimes described as turning the application inside-out as compared to function-oriented interfaces. The main focus of the interaction changes to become the users' data and other information objects that are typically represented graphically on the screen (...). Users achieve their goals by gradually massaging these objects (...) until their state, as shown on the screen, matches the desired result."* (*Nielsen 1993: 59*).

In an OOUI-based operating system such as IBM's Common User Access Workplace (*Berry and Reeves 1992*), all objects and their functionality co-exist in a single integrated and consistent graphical work environment that is not fragmented into many different walled applications with individual interaction styles or storage formats like today's application-oriented WIMP desktop (*Jetter, König, Gerken et al. 2008*). The functionality and behavior of the UI is not defined by many different applications and their application structure, but by the available classes of objects inside the workspace, which properties they have, and which operations can be issued on them. Nielsen writes *"The main difference is that the user no longer needs to think in terms of running applications, since the data knows how to integrate the available functionality in the system. In some sense, such an object-oriented system is the ultimate composite editor, but the difference compared to traditional, tightly integrated multi-media editors is that the system is open and allows plug-and-play addition of new or upgraded functionality as the user desires without changing the rest of the system"* (*Nielsen 1993: 64*). Also, OOUIs have fewer objects and more reuse of the same objects in many tasks whereas typical WIMP GUIs require many applications, i.e., one per task (*Mandel 1994: 229*). This idea of more reuse, and thus also consistency, is similar to Raskin's hypothetical "Humane Interface" that strives for *unification* and is not based on applications but views software as a modular

set of *commands* and *transformers* that can be flexibly applied to all the different objects on the user interface (*Raskin 2000: 139-48*).

3.1.3 OOUIs and Post-WIMP Interaction

Working with an OOUI is different from working with an application-oriented WIMP UI where the users' assumed work procedures become manifest in many specific applications and their individual sequences of pages, menus, dialogs, and widgets. In contrast, OOUIs have a *"flexible structure-by object"* instead of the *"rigid structure-by function"* of application-oriented GUIs (*Mandel 1994: 229*). While in application-oriented user interfaces *"users must follow the application structure"*, OOUIs enable users to *"perform tasks in their own way or innovate"* (*Mandel 1994: 229*). As mentioned, this flexibility of an OOUI is lauded for making user interfaces more expressive and empowering and for treating the user more like a problem-solver and less like a dumb process-follower (*Pawson, Bravard, and Cameron 1995*; *Pawson 2004: 35*). This also resonates with Kidd's (*Kidd 1994*) and Collins' discussion of the *production work* of the past versus present-day *knowledge work* and how this demands for the *variety* and *flexibility* of OOUIs instead of *repetition* and *rigid structure* (*Collins 1995: 29*).

This flexibility of OOUIs is also an advantage in light of the *situatedness* (*Suchman 1987*) or *embodiment* (*Dourish 2004*) of users' actions in a post-WIMP world where computational power is deeply woven into the fabric of our physical and social work environment. For Suchman, human action is constantly constructed and reconstructed from dynamic interactions with the world. As Dourish summarizes, Suchman in her work *"provided detailed analyses of the interactional problems arising from the mismatch between, on one hand, the clean-cut, abstract and stable models that a system might have of interaction and, on the other hand, the much more messy, immediate, and fluid circumstances in which the system's users find themselves"* (*Dourish 2004: 73*). Therefore, trying to design a post-WIMP interactive system that achieves the user's goal by implementing a single predefined plan or script with the assumed sequence of necessary actions often fails. Studies from ethnography document many of such failures to anticipate sequences of action and also demonstrate that order – and hence sequence – is an ongoing product of people's work (*Robinson 1993*). Given the past successes of Tayloristic scientific management, the search for abstracted optimum sequences is understandable, but in practice users do not execute such plans or follow sequences. Instead, their actions are an ongoing, improvised activity with moment-by-moment responses to the situations in which the users find themselves (*Dourish 2004: 72*). As a consequence, an interactive system should be designed for *unanticipated use* (*Robinson 1993*) and should be *expressive* enough to empower users to act flexibly and to improvise when needed (*Pawson, Bravard, and Cameron 1995*).

In the past, the WIMP desktop metaphor with its windowing paradigm, multi-tasking, composite documents, and direct manipulation achieved this to some degree by making multiple applications simultaneously visible and accessible. However, today's WIMP desktop suffers from the two *"dueling interaction models of personal-computing and web-*

computing" with *"a severe negative impact on human-computer interaction"* (*Müller-Prove and Ludolph 2007*). Many present-day applications on the World Wide Web have adopted a strictly sequential, form-based, and conversational interaction style, for example during a checkout process. The application's functionality is provided in virtual pathways through the application space, e.g., page flows or series of dialogs. Thereby the conversation between user and system is intermittent and based on discrete input events such as filling out forms and clicking widgets or links without direct manipulation. Instead of a model-world, such interfaces resemble semi-automated conversations along navigational paths that guide users through the necessary processes and information resources for achieving their goals. Often Web users cannot react to the messy circumstances of the real world, e.g., frequent shifts in user roles or working styles, momentarily not having all necessary information at hand, or having to decide on some aspects later. Furthermore, Web-based storage and the local file system are separated. Transferring objects or files between them is often time-consuming and suffers from different storage formats. When moving into the post-WIMP world, further problems arise, since the operating systems or shells of post-WIMP devices (e.g., smart phones, tablets, interactive tabletops) often have no or only a very rudimentary support of multi-tasking, window management, or copy & paste between "apps" or applications. This current trend towards *"dumbing down interaction"* with apps is criticized by Beaudouin-Lafon[20], who calls for rethinking application-orientation when moving to post-WIMP or ubiquitous computing (*Beaudouin-Lafon 2000*; *Klokmose and Beaudouin-Lafon 2009*).

My claim is that designers of post-WIMP multi-user interactive spaces should therefore focus on the objects of collaboration and provide them as first-level citizens of an OOUI. Users should not be forced to follow a rigid application structure or static work procedures. Furthermore, users should not be concerned with application boundaries, different storage locations, and formats. While page-oriented "Web-style" designs or applications with sequential work processes and step-by-step interaction can achieve great usability when automating a finite number of clearly defined tasks or business processes, e.g., in e-commerce, they are not appropriate in the context of ill-defined tasks such as collaboratively interacting with personal and shared information in ubiquitous computing environments. When using OOUIs instead of application- or function-oriented UIs, the objects of collaboration become the first-level citizens of the UI and thus OOUIs achieve a greater flexibility when contexts and working styles are changing and overcome the boundaries and inconsistencies between different applications, storage locations, and formats. This is particularly useful for the domain of post-WIMP collaborative knowledge work, as I discuss in the following section.

[20] See talk "Of Tools and Instruments" by Michel Beaudouin-Lafon at Dagstuhl Seminar 12351 "Interaction Beyond the Desktop". *http://www.dagstuhl.de/mat/Files/12/12351/12351.BeaudouinLafonMichel1.Slides.pdf* (Accessed Nov 5, 2012).

3.1.4 OOUIs for Knowledge Work

As mentioned, OOUI design begins with understanding and analyzing the targeted domain and typical tasks. The definition of knowledge work from section 1.5 can serve as a generic starting point. During a top-down design process, the generic higher level activities or workflows of knowledge work can be contextualized for the specific target domain and can be hierarchically decomposed into lower level task models, e.g., essential use cases or scenarios. These task-oriented models can then be used to increasingly flesh out the details of the user interface design, starting from its navigation map and wireframes, and ending with the visual design of pages or dialogs. Such a task-oriented top-down design from the abstract to the concrete has become a standard procedure in interaction design and usability engineering, e.g., in the usage-centered design methodology (*Constantine and Lockwood 1999*).

However, in the case of collaborative knowledge work by co-located users in a multi-device environment, e.g., during a brainstorming session or collaborative sketching, it is not sufficient to consider the users' interaction as a task-oriented sequential process that can be supported by a single hard-coded sequence of interactions. Instead, collaborative knowledge work in a multi-device interactive space is a distributed, concurrent, and sometimes seemingly chaotic activity that follows higher level goals or workflows and not predefined sequences of lower level interactions or rigid application structures. The users' actions are situated in a changing physical, social, and technological environment with multiple devices, in which multiple users at multiple points of actions constantly pick up, use, manipulate, recombine, create, and destroy information objects without following clearly defined processes that terminate at clearly defined goals. In other terms, collaborative knowledge work is *mixed-focus collaboration* (see section 1.4.3, p.13): The user goals and roles during knowledge work change and shift constantly, since encountered information items or intermediate group results might change the entire nature of a user session, its goals, and the roles and coupling of individual users therein.

Furthermore, post-WIMP tangible interaction and multi-touch input transform the sequential nature of interaction at the interface. *"The single point of control that traditional interfaces adopt leads naturally to a sequential organization for interaction– one thing at a time, with each step leading inevitably to the next. (...) When we move from traditional models to tangible computing, sequential ordering does not hold. It is not simply that interaction with the physical world is 'parallel' (a poor mapping of a computational metaphor onto real life), but that there is no way to tell quite what I might do next, because there are many different ways in which I might map my task onto the features of the environment"* (*Dourish 2004: 51*).

Designing UIs for such complex and ill-defined tasks only based on sequential task models is difficult or even impossible. *"This does not mean that CSCW systems design cannot be informed by analysis of practice: only that the practice is better conceptualized in a multi-dimensional space rather than [only] as temporal task sequences"* (*Robinson 1993*). For Robinson, a further important dimension of conceptualizing practice, are the

"common artifacts" or objects that the collaboration is based on. This can help to achieve what Robinson calls a *"design for unanticipated use"* (*Robinson 1993*).

Such a perspective can be a valuable complement to task-oriented analysis and modeling, particularly when regarding the inherently object-oriented nature of *information items* in knowledge work (*Jetter, König, Gerken et al. 2008*; *Jetter, Gerken, Zöllner et al. 2010b*): Jones and Teevan introduce the *information item* as a fundamental concept for the consideration of personal information management (*Jones and Teevan 2007: 7*): An information item (e.g. a real world printed document or handwritten note, an email message, a web page or a reference to a web page) is a *"(...) packaging of information in a persistent form (...)"* and with an associated *information form* or *information type*. Items have forms such as "paper document", "electronic message", "web page" or "web bookmark" and this information form is *"determined by the constellation of tools and applications that make it possible to manipulate (acquire, create, view, store, etc.) the item"* (*Jones and Teevan 2007: 7*).

This notion of an *information item* can also be expressed in object-oriented terminology (*Jetter, König, Gerken et al. 2008*): information items are instances of different classes of information forms. These classes have attributes in which the individual metadata and content of an instance is stored. The different manipulations are defined by the functions and behaviors of the class, e.g., if a class of items can be moved between different locations, the class should define a *Move()* function. If other items can be dropped on top of it, it should define a *ReceiveDropItem()* function, etc.

Jones and Teevan have also observed that although a person's interactions with an information item vary greatly depending upon its form, there are many essential similarities in the way people interact with information items (*Jones and Teevan 2007: 8*). Object-oriented analysis and modeling can be used to analyze the differences and commonalities of information forms from a user's perspective. Using OO features such as *inheritance*, *generalization*, and *polymorphism*, it is possible to systematically catch all the essential similarities and differences for the user in common base types or classes. This helps to iteratively reduce the complexity of the model. *"Polymorphism can be used very effectively in OOUIs to reduce the number of commands or operations the user needs to learn and remember"* (*Collins 1995: 91*).

The resulting class hierarchy reveals all the properties of information items and all the ways to manipulate them that are relevant for the user and the user interface design. Thus OO mechanisms facilitate the systematic integration of very different types of information into a single model while preserving a maximum degree of consistency for the user. As discussed by Collins, this is important for successful OOUI design: *"higher level issues must be considered at the same time. Interactions should be consistent across objects of the same class; where possible, operations should be polymorphic - applicable to different object types. This reduces the number of interaction behaviors and simplifies the interface"* (*Collins 1995*). The resulting model can then be used as a basis for designing behavior, functionality, and appearance of information items inside the interactive

space. If properly applied this leads to a coherent and consistent behavior throughout the entire model-world UI that also makes it easier for users to apply previous experiences for discovering new functionality and taking out new kinds of tasks. *"Carefully defining and clearly distinguishing object classes enables users to learn and predict the behavior of objects"* (*Roberts, Berry, Isensee et al. 1998: 16*).

Figure 42 shows an example object model that I created for the study discussed in the following section 3.2. It represents a ZOIL-based OOUI for searching, browsing, and annotating hotels from a hotel catalog. Therein, the UI objects *Image*, *Comment,* and *Hotel* are inheriting basic attributes and behaviors from their common base class *ContentItem*. Thus these items share their standard behaviors with regard to moving, rotating, and resizing. Furthermore, they share the ability to create visual links between them. Users do not have to differentiate between these classes of objects when performing these operations on them.

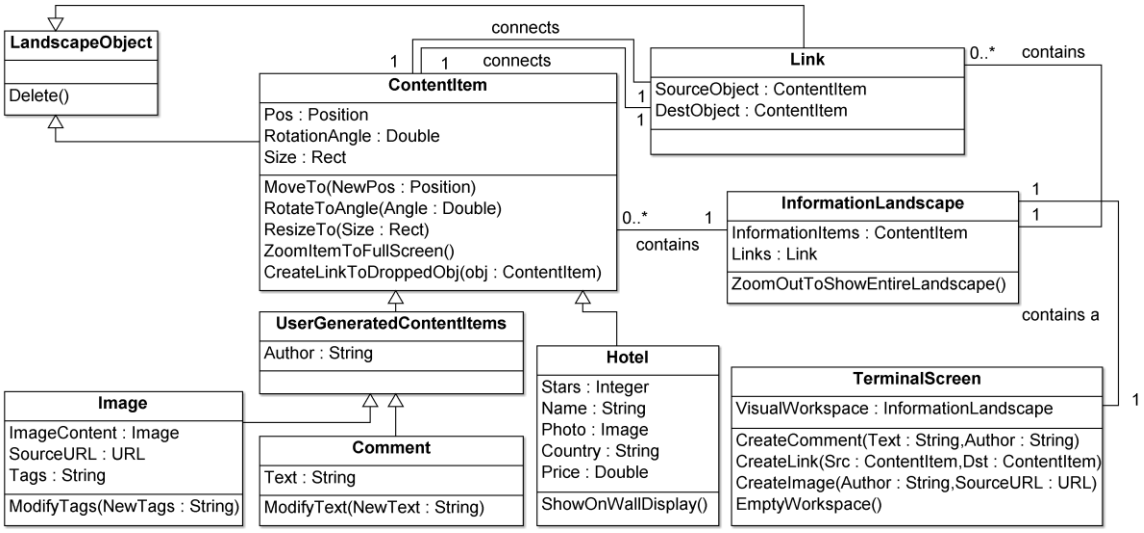


Figure 42 – An example of an object model of a ZOIL-based OOUI for searching, browsing and annotating hotel data from a hotel catalog. This was created for the case study discussed below and in (*Jetter, Gerken, Zöllner et al. 2010b*).

3.2 A Case Study of OOUI Design and Modeling with ZOIL

While the previous theoretical discussion of OOUI properties reveals many advantages that OOUIs could have for designing post-WIMP interactive spaces, there is only very few research on how well OOUI design and modeling can be applied in practice. Although some articles, chapters, and books have addressed WIMP OOUIs since the 1990s (*Beck, Janssen, Weisbecker et al. 1994*; *Berry and Reeves 1992*; *Collins 1995*; *Haywood 2009*; *Mandel 1994*; *Nielsen 1993*; *Pawson and Matthews 2001*; *Pawson 2002*, *2004*; *Roberts, Berry, Isensee et al. 1998*), there are only three publications that propose a clearly outlined OOUI methodology and modeling technique and claim to have applied their methodology in practice to gain empirical data from actual use:

First, in 1994, Beck et al. introduced the TASK methodology for integrating OO analysis into graphical user interface design for desktop systems (*Beck, Janssen, Weisbecker et al.*

1994). During TASK's analysis activity, a task model and an initial object-oriented *object model* is built, which is then refined in another activity to an object-oriented *application specification*. This specification is used as a *conceptual user interface model* during user interface design and the views, dialogs, and the actual screen representations of conceptual objects are derived from it. Beck et al. report about a successful application of TASK and its supporting tools for the design of insurance and production planning systems. However, there is no description of how the empirical data was collected and what "successful" means in detail. There is no description of the employed tools, intermediate results, and the amount and kind of human intervention for translating the *conceptual user interface model* into concrete user interface design and its implementation.

Second, in 1998, Roberts et al. introduced IBM's OVID (Object, View, and Interaction Design) methodology to bridge user interface design and software engineering by using the notation and modeling techniques of successful code design methodologies and combining these with user interface design and usability engineering processes (*Roberts, Berry, Isensee et al. 1998*). At the heart of OVID is the *designer's model* whose main component is the *object model*, an object-oriented model similar to a UML class diagram that includes descriptions of the objects users will employ to perform their tasks, the properties of those objects, and the interrelationships between them. Textual and formal notations of tasks (e.g., use case diagrams) can be used to identify those objects that users have to act on and that should be provided to them on the user interface. Roberts et al. show illustrations and example artifacts from a case study of using OVID to create the IBM RealCD CD player application. However, they do not report about the difficulties and pitfalls that they encountered during applying OVID and OOUI methodologies and the presented material is more intended for teaching the method than for critically reflecting about its practical value.

Third, in his dissertation about Naked Objects, Pawson provides a deep insight into his OOUI methodology and framework (*Pawson 2004*). This also includes a chapter with a case study of the development of a new benefits processing system for the Irish government. The case study started with a one-day exercise involving a group of managers to identify the core objects of the application domain without the attempt to specify use-cases up front. The result was a list of core classes for which the group was asked to imagine how users would interact with each object resulting in a very rough definition set of behaviorally-rich objects. This was translated into a visual mock-up that was used to demonstrate a small set of specific use-cases. The positive reactions of user representatives resulted in further explorations of the concepts and ultimately led to a contract for replacing the old system with the new OOUI system by 2002. While this is a success story of OOUI approaches in practice, Pawson does not report in detail about the difficulties of using object-oriented modeling and design within the groups of user representatives, designers, or developers. Instead he focuses on the impact the resulting system had on the organizational level and individual users, e.g., increased usability/operational agility.

3.2.1 Study Design

In order to learn about the applicability of OOUI design and modeling in practice, we[21] conducted a small 3-week case study with 11 participants (9 graduate-level and 2 undergraduate students of computer science). The question guiding the study was how well participants can apply OOUI design and modeling techniques during a small-scale project. The participants were divided into five teams (4 teams with 2 members, 1 team with 3 members). Each team was given the same task with identical requirements for designing and implementing a ZOIL-based user interface for search, annotation, and sensemaking of hotel data and hotel photos in a zoomable information landscape. Due to the limited timeframe of the study, the task's complexity was restricted to create a UI that can be operated by a single user with mouse or with a digital pen on a large 67" vertical screen with HD resolution. The employed design and modeling techniques are described in detail in (*Jetter, Gerken, Zöllner et al. 2010b*) where the results of the study were published as a contribution to the field of human-centered software engineering.

In a first one-hour session we presented the OOUI modeling approach to all teams. As an example we created and explained an object-oriented model of a ZOIL user interface in a UML-like notation like in Figure 42 for accessing a fictitious image database. Starting from this example, the teams were then given the assignment to create an own model for a different ZOIL user interface until the next session in two weeks. The new user interface to model should allow users to explore and discuss hotels from a hotel database. As input for their modeling and design activity, the teams were handed 8 informal functional requirements (e.g., *"user must be able to add a textual comment to the workspace"*, *"user must be able to modify tags of an image"*) and a list of 22 required object properties (e.g., *"each <Comment> has an <Author>"*, *"each <Image> carries <Tags>"*) together with an explanation of the raw data contained in the database. As experimenters we used the same input to create a solution model (see Appendix 9.1) that we used to check the participants' results for completeness and formal correctness.

Two weeks later, we carried out individual one-hour team sessions during which each team completed three tasks: First, each team presented and explained their resulting model. Then we asked the team to check if their model really supports the 8 functional requirements by carrying out a walkthrough. We then presented the team our own solution model and asked them to validate this unknown model by another walkthrough. After this, each team member filled out a questionnaire to rate the difficulty of the three tasks using 5-point scales with semantic differentials, e.g., *"For modeling the user interface, you were provided with user tasks and a description of the raw data. How do you rate the difficulty of creating your model based on this input? (1: very difficult, 5: very easy)"*.

[21] Any use of "we" in section 3.2 refers to me (Hans-Christian Jetter), Jens Gerken, and Michael Zöllner. The study was conceived of and designed by me. I also designed the study tasks, questionnaires, and materials. Jens Gerken and Michael Zöllner assisted me during conducting the study with participants of our course on visual information seeking systems. Jens Gerken also provided the statistical analysis of and the figures for the results of the questionnaires. The study was published in a full paper for HCSE (*Jetter, Gerken, Zöllner et al. 2010a*) that I authored and that is reproduced in parts here.

At the end of the second sessions, the teams were instructed to design and implement a user interface with the ZOIL software framework with C#/WPF based on a reduced version of the solution model until the final session in the following week. For time reasons, the reduced model did not contain the functionality for resizing, rotating, and connecting objects with visual links. In these last sessions, each team individually presented the current state of their interactive prototype and filled out a further questionnaire to rate the overall usefulness of the modeling approach during their project and the difficulty to apply it on user interface design and implementation.

As reflected in the study design, the primary goal was to collect qualitative interview data from a realistic project, instead of measuring quantitative data (e.g., task times, task completion rates) in a controlled experiment. In addition, the questionnaires helped to provide a first assessment of the applicability of OOUI design and modeling approaches for a ZOIL-based user interface.

3.2.2 Study Results

In summary, the results of the study revealed that participants were able to apply OOUI design and modeling techniques for the given task. All teams presented models that were formally correct and supported the 8 functional requirements. All teams were able to carry out a walkthrough to validate their own and an unknown model. At the end of the course during which the study took place, all five resulting interactive prototypes carried the required functionality specified in the model and – based on my own assessment – 4 out of 5 achieved a sufficient level of usability and user experience. More details about the resulting prototypes are presented in section 3.5.2 (p.87).

Figure 43 shows the results from the questionnaires: the creation of a model (mean=3.45, SD=0.93) and checking the own or someone else's model with a walkthrough (mean=3.1, SD=1.05 and mean=2.9, SD=1.14) were considered neither particularly difficult nor easy. This can be regarded as a case in favor of the approach, since the students were given only a very brief introduction without an extensive training phase. Furthermore, the overall utility of the modeling technique was considered as useful (mean=4.1, SD=0.99) by the participants after the implementation of the prototype.

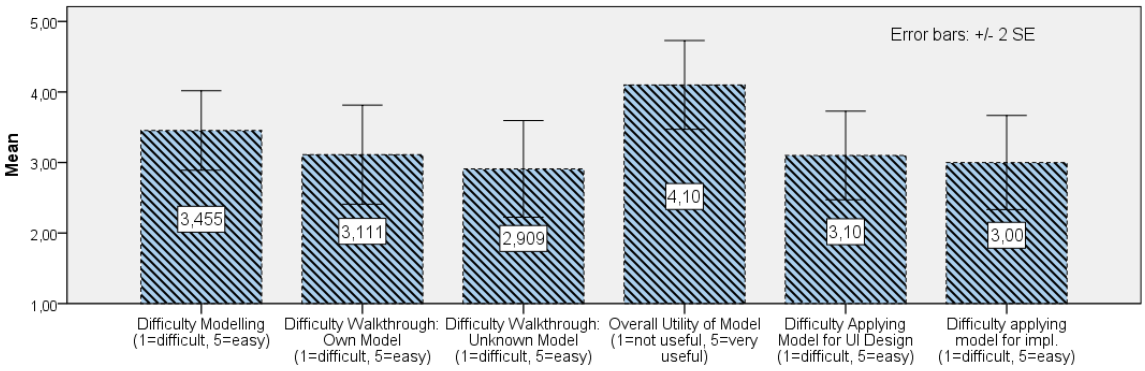


Figure 43 – Results from the questionnaires on the applicability of ZOIL's OOUI design and modeling approach. Source: (*Jetter, Gerken, Zöllner et al. 2010b*)

As is also reflected in the neutral rating for the difficulty of applying the model for UI design (mean=3.1, SD=0.99), some problems occurred because of the unfamiliar use of UML class diagrams to model the functionality of a user interface instead of code objects. During the first and second session, participants repeatedly reported that they sometimes had fallen back into the more familiar modeling of code objects and lost track of their original intention to model the user interface objects from a user's perspective. However, the participants reported that they got increasingly used to the approach and then found it useful to support the design and implementation.

Similarly, the use of the model for UI implementation also received neutral results (mean=3.0, SD=1.0). This can be explained by the non-trivial task of mapping UI objects from the OO model to their actual representations in C# code. This problem also surfaced in the study of the ZOIL software framework’s API usability in connection with the *Model-View-ViewModel* pattern and is discussed in greater detail in section 3.5.7 (p.102).

In conclusion, regarding the unfamiliar use of object-oriented modeling and design for user interfaces and the minimum amount of training, we consider these results as evidence for the applicability of OOUI techniques to the design of ZOIL-based user interfaces. Although originally introduced for traditional WIMP/GUI interaction, student designers and developers used OOUI design and modeling techniques successfully to create object-oriented models of ZOIL-based user interfaces that went beyond traditional GUI interaction techniques and made use of zoomable user interfaces with semantic zooming and information visualization (see examples in section 3.5.4). While the participants considered OOUI techniques useful for their design and implementation, the neutral ratings of difficulty reveal that the techniques are not considered easy to apply without prior practice, so that there is a need to train designers and developers accordingly to make them learn the required skill set for “thinking in objects”.

3.3 P#1: “Provide Post-WIMP Functionality as Objects, not Applications.“

As a summary of the first part of this chapter, I formulate the *1st ZOIL design principle* about the role of OOUIs for ZOIL as follows:

“Provide post-WIMP functionality as objects, not applications.”

This includes following considerations about object-oriented vs. application-oriented interaction:

1.) Collaborative knowledge work in a multi-user, multi-device interactive space is a distributed, concurrent, and sometimes seemingly chaotic activity that follows higher level goals or workflows and not predefined sequences of lower level interactions or rigid application structures. The user goals and roles during knowledge work change and shift constantly, since encountered information items or intermediate group results might change the entire nature of a user session, its goals, and the roles and coupling of individual users therein. Therefore users should not be forced to follow a rigid application structure or static work procedures. Furthermore, users should not be

concerned with application boundaries, different storage locations, and formats. Instead of thinking in processes and sequential application structures, designers of post-WIMP interactive spaces should focus on the objects of collaboration and provide them as first-level citizens of an OOUI.

2.) An OOUI focuses the users on objects – the "things" people use to accomplish their work (*Roberts, Berry, Isensee et al. 1998: 11*). There is a flexible structure-by object instead of a rigid structure-by function. Users may perform task in their own way or innovate (*Mandel 1994*). Users see objects instead of applications (*Mandel 1994: 241*). There don't seem to be identifiable applications in an OOUI; there is merely a constellation of objects that a user can consider and manipulate (*Tibbetts 1991*).

3.) OOUIs are based on the hypothesis that *object-orientation*, i.e., understanding, analyzing, and modeling the world in terms of *objects* and their *classes*, is a natural and very efficient way of conceptualizing our environment and is close to the way we reason in our everyday world (*Collins 1995: 77*; *Søgaard 2006*). This analysis and modeling of a specific problem domain from the real-world in terms of objects, classes, attributes, functions, behaviors, and relations is at the heart of object-orientation and is the *meta-model* (*Collins 1995: 70*) that OOP and OOUIs share. Using OO features such as *inheritance*, *generalization*, and *polymorphism*, it is possible to systematically catch all the essential similarities and differences of UI objects in common base types or classes. This helps to iteratively reduce the complexity of the model and simplifies the interface (*Collins 1995*).

4.) Initial studies with designers and developers revealed that OOUI techniques are considered useful but are not easy to apply without prior practice. There is a need to train designers and developers accordingly to make them learn the required skill set for "thinking in objects" and to solve typical problems such as the non-trivial mapping of UI objects to their representations in code.

3.4 OOUI Implementation with the ZOIL Software Framework

As discussed in section 1.6.2 (p.21) of the introduction, implementing a multi-user, multi-device, post-WIMP interactive space for object-oriented interaction is a challenging task. Therefore this thesis introduces the ZOIL software framework for Microsoft's .NET platform and Windows Presentation Foundation (WPF) to support developers during implementation. The ZOIL framework is an extensible collection of components and classes written in C# and WPF's declarative XML-based UI markup language XAML ("Extensible Application Markup Language") with approximately 7,000 lines of code and is shared for reuse in academia and industry as open-source under the BSD License[22]. It provides all the necessary core functionality for implementing ZOIL-based OOUIs for a multi-user and multi-device interactive space. Figure 44 gives an overview of the functionality of the ZOIL framework and its different software components.

As illustrated, the ZOIL framework serves as a kind of middleware between the ZOIL client applications on the top, and the lower level APIs and operating system at the bottom. It provides high-level functionality in the three areas of *presentation & interaction*, *communication*, and *persistence & synchronization*. These key aspects of ZOIL-based interactive spaces can be realized by using existing software components from the ZOIL framework without the need for extensive knowledge about the details of the underlying lower level libraries or APIs.

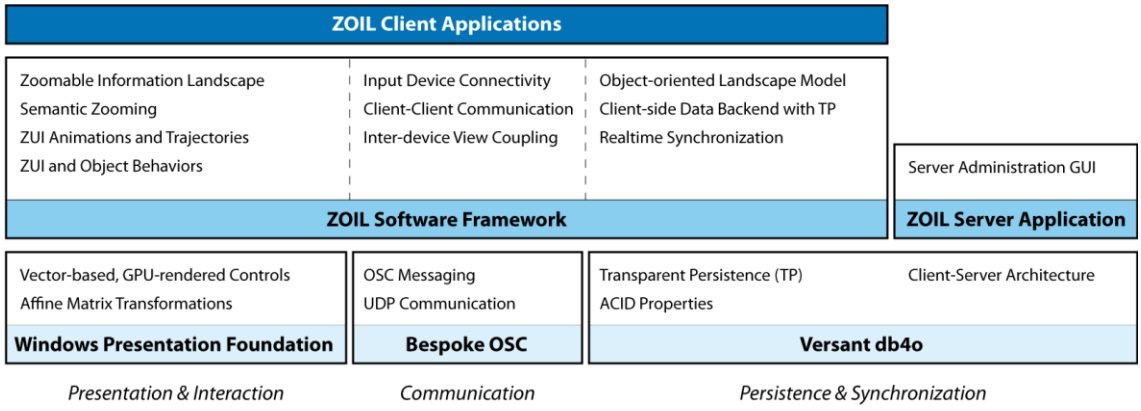


Figure 44 – The ZOIL software framework sits between the individual ZOIL client applications at the top and lower level APIs at the bottom. It also provides a ZOIL server application with GUI. Source: (*Jetter, Zöllner, Gerken et al. 2012*).

In the light of the framework's volume and scope, the next sections do not describe the framework's implementation and architecture in all details, but focus only on its most defining characteristics and the most important software patterns that were used during its development. Appendix 9.2 contains additional material describing the framework's architecture and functionality.

[22] ZOIL Framework at Codeplex.com: *http://zoil.codeplex.com* (Accessed Jun 15, 2012).

The complexity of the framework made it necessary to include student developers to support its implementation: Andreas Engl contributed a first implementation of the zoomable information landscape and the semantic zoom according to my specifications of their architecture and functionality. Furthermore, with the *EuroITV* demonstrator (see section 4.3.2, p.122 and section 4.3.3, p.125), he provided the first interactive prototype that used the ZOIL software framework in his Bachelor's thesis (*Engl 2008*; *Jetter, Engl, Schubert et al. 2008*). Michael Zöllner contributed the first implementation of the framework's data model and data backend in his Bachelor's thesis (*Zöllner 2009*). This was based on my specification of a client-server architecture using the db4o object database to achieve transparent persistence and real-time synchronization. He also suggested and integrated WPF's *Model-View-ViewModel Pattern* to connect the framework's data model to the presentation layer, i.e., the information landscape.

3.4.1 Implementing a Shared OOUI Object Space

During runtime, any OOUI needs an underlying object-oriented data model or *object space* that contains the class definitions and the classes' individual instances as objects. Furthermore, this object space also implements the rules and interactive behaviors of the object-oriented model-world UI and stores the current state of the entire model-world and its objects. Due to the similarities between OOP and OOUI, it is a straightforward task to create such an object space with an object-oriented programming language such as C#. In principle, the implementation is a straightforward conversion of a formal model such as Figure 40 (p.56) or Figure 42 (p.64) into OOP class definitions and code. However, for introducing an OOUI into a ZOIL-based post-WIMP interactive space, the object space has to be shared across device boundaries and thus has to expose two further properties that involve greater effort for implementation: *transparent persistence* and *real-time synchronization*.

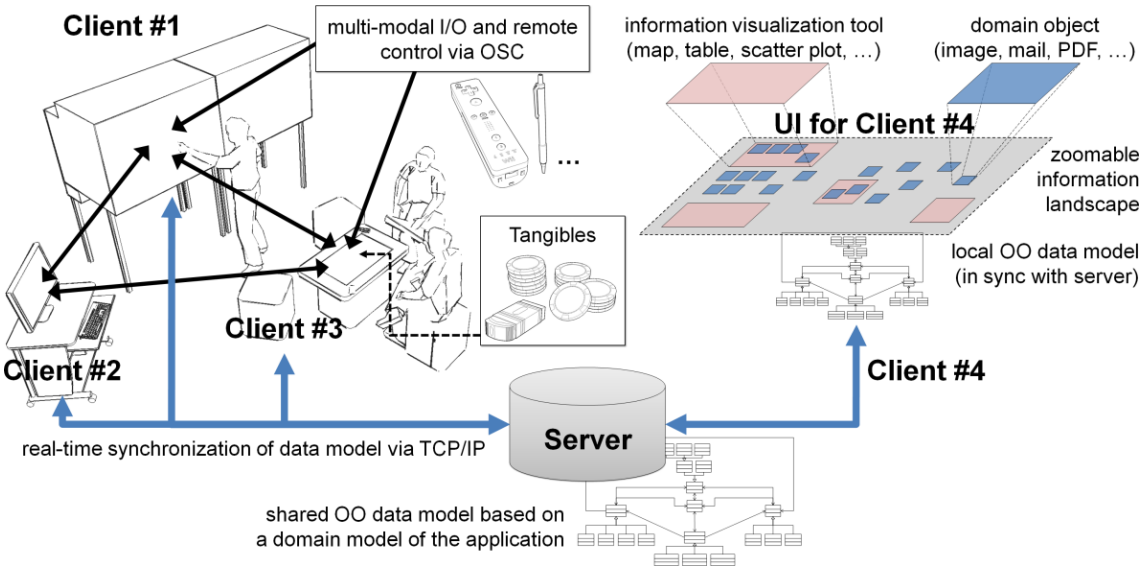


Figure 45 – In a ZOIL interactive space, the persistent object space is distributed in real-time across multiple client devices and/or users. The right section shows how a ZOIL client visualizes the workspace's data model as a zoomable information landscape and in visualization tools. (see chapter 4). Source: (*Jetter, Zöllner, Gerken et al. 2012*)

3.4.2 Transparent Persistence

Since ZOIL is intended for multi-device, multi-display, and multi-user scenarios, all the different devices and their application software inside an interactive space must be able to communicate and cooperate with each other. To provide the required technological environment for this communication and cooperation, the ZOIL software framework enables developers to easily create a *shared object space* that contains all objects in the information landscape including their states.

Since the lifetime of an object in this shared object space has to be independent from the local lifetimes of objects in the devices, it is necessary to store the object space *persistently* at a central location, i.e., the ZOIL database server, to which the ZOIL applications that are running on the different devices connect as ZOIL clients. All objects stored on this ZOIL server are persistent and keep their state and properties after the end of a user session or unexpected shut downs. Figure 45 illustrates this architecture.

The ZOIL software framework provides this infrastructure and the necessary base classes to the ZOIL application programmer in the form of a client-side data backend API. It lets developers easily access and modify the persistent object space that resides on a central object-oriented ZOIL database server. The ZOIL server is implemented using the C#-API of the embedded object database db4o[23]. It uses this API to store and retrieve binary representations of the information landscape and the contained objects during runtime. The data is physically located in a binary db4o database file in the file system of the server that contains all information about the information landscape with all objects and their states. Thereby db4o guarantees the necessary ACID properties[24] for all database transactions between ZOIL clients and the ZOIL server.

Such a concept of a shared object space that is distributed over multiple devices for achieving synchronous collaboration is not specific to ZOIL. For example, the COAST[25] software framework that was published as open source and used in the context of Streitz et al.'s i-LAND (*Streitz, Geißler, Holmer et al. 1999*) and Tandler et al.'s ConnecTables (*Tandler, Prante, Müller-Tomfelde et al. 2001*) used a similar approach. However, the framework was discontinued in 2003 and it was implemented in the Smalltalk programing language (*Goldberg and Robson 1983*). This is problematic for present-day post-WIMP scenarios, because it is difficult or even impossible to achieve interoperability between legacy Smalltalk code and present-day platforms such as .NET/WPF or APIs for post-WIMP input and output such as the Microsoft Surface SDK.

In contrast, ZOIL tries to provide a low-threshold persistency and synchronization that is seamlessly integrated into Microsoft's .NET/WPF/Surface ecosystem and is also ideally transparent to the application programmer. To achieve this, ZOIL's client-side data backend uses db4o's API for *transparent persistence*. This enables C# programmers to

[23] db4objects by Versant: http://www.db4o.com (Accessed Apr 27, 2012).

[24] ACID (Atomicity, Consistency, Isolation, Durability): A set of properties that guarantee that database transactions are processed reliably.

[25] The openCOAST distribution site: *http://www.lifia.info.unlp.edu.ar/~casco/opencoast/* (Accessed Apr 27, 2012).

access and change persistent data with the same ease as non-persistent local data models in main memory. Programmers can simply change the states and properties of objects locally on the client-side using standard C# code without explicitly sending and receiving database queries to the server. Transparent to the programmer, ZOIL's data backend observes and collects all these changes to the objects in main memory and transmits them in a bulk via TCP/IP to the central server in regular intervals (typically 100ms). As described in the following section about real-time synchronization, the server then informs all other clients about the changes, so that the other clients' data backends can retrieve and execute them within 100-300ms in typical setups.

Using transparent persistence also facilitates iterative design due to its object-oriented nature: Unlike with relational databases, C# code changes in object definitions are automatically detected using reflection during runtime without the need for manual updates of object-relational mappings or table schemata. This is an advantage of ZOIL compared to using SQL[26] databases or solutions such as Hibernate[27].

3.4.3 Real-Time Synchronization

All changes to the object space made with one device must be synchronized and propagated to all other active devices and their application software in real-time to allow for synchronous multi-user and multi-device collaboration. ZOIL's client-side data backend and the server-side database ensure such a real-time synchronization of changes that enables co-located synchronous collaboration across device boundaries.

To achieve real-time synchronization, all changes made to an object on one client are immediately synchronized via the server with all other connected clients. ZOIL uses a simple non-blocking synchronization scheme in which the latest change made to an object always wins. Although this can introduce conflicts or change races during concurrent user manipulations (e.g., concurrent dragging of the same object by different users with different devices), such problems were not observed in practice for ZOIL's example prototypes or during user studies. Instead, users applied their familiar *social awareness and skills* as described in the Reality-Based Interaction framework (*Jacob, Girouard, Hirshfield et al. 2008*) to coordinate their actions, e.g., natural social protocols, verbal and gestural communication, and group awareness[28].

To more reliably avoid any conflicts or change races between different users by technological means, the ZOIL framework also offers a locking mechanism. This mechanism can be used by a ZOIL-based application to temporarily lock objects during user manipulations. Locked objects are exclusively interactive for the user who first started a manipulation and remain non-interactive for all other users until this

[26] SQL (Structured Query Language) is the leading programming language for managing data in relational database management systems.

[27] Hibernate is a popular object-relation mapping library for achieving persistence. *http://www.hibernate.org* (Accessed Apr 23, 2012).

[28] Chapters 5 and 6 discuss how to design ZOIL user interfaces for the users' social awareness and skills in greater detail.

manipulation is finished. This provides atomicity not only on the database level but also on the level of user manipulations and can be helpful in future scenarios with a need for highly synchronous safety-critical collaboration.

3.4.4 Model-View-ViewModel Pattern

While transparent persistence offers low-threshold persistency for an object's content and state, it cannot be applied directly on the user controls and the visual hierarchies of WPF. Moreover, such exact replications of user objects, widgets, or controls would not be desirable in ZOIL, since each client should provide the user with an independent view of the shared information landscape, so that users can independently navigate and manipulate content and are able to make use of device-specific display and interaction capabilities, e.g., different screen resolutions, multi-touch input, pen input.

For this reason, the ZOIL framework uses the *Model-View-ViewModel* (MVVM) architectural pattern[29] to provide a Model-View-Controller- or MVC-style (*Goldberg 1990*) separation between the persistent data *model* of an object in the object space and the non-persistent *view* of the object on the UI of an individual device: Each object's *model* is shared via the server with all others clients, but the corresponding *view* is not shared and only resides locally on the client-side. As soon as a *model* is created (or destroyed) in a local or remote client, the corresponding device-specific *views* on all other clients are created (or destroyed) accordingly.

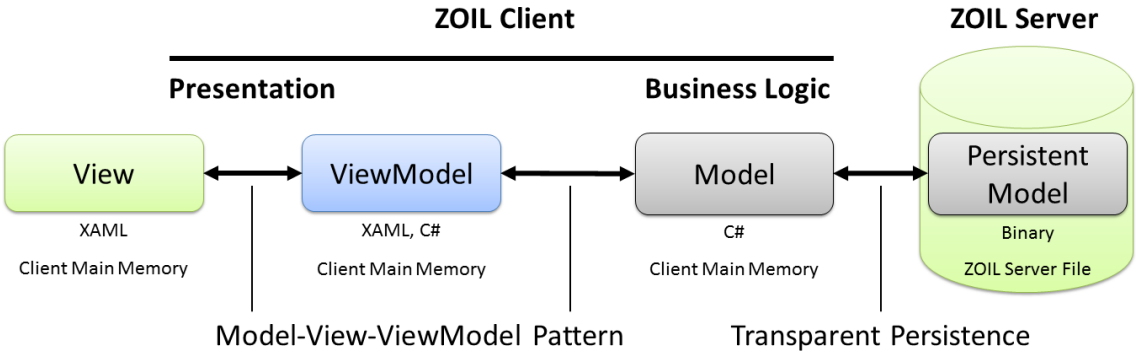


Figure 46 – The *Model-View-ViewModel* pattern is used by ZOIL client applications to separate the C# business logic in an object's data model from the declarative definition of appearance and behavior of an object's view in XAML. *Transparent Persistence* observes all client-side changes that are made to the model in main memory and makes them persistent in the binary file of the ZOIL server.

The key advantage of using MVVM in ZOIL over using a traditional MVC approach lies in the *ViewModel*, which plays the role of an adapter between the visual appearance and interactive behavior of the *view* and the underlying C# code with the business logic of the *model*. The *ViewModel* acts as an interpreter between *model* and *view*. Using WPF's two-way data binding and automatic converters, changes in the actual raw data in the *model* are automatically translated into human-readable formats and the *view* is instantly

[29] An excellent introduction to the Model-View-ViewModel Pattern by John Gossman can be found at http://blogs.msdn.com/b/johngossman/archive/2005/10/08/478683.aspx. (Accessed June 14, 2012).

updated. Similarly, input events and data that are received when a user interacts with a *view* or enters human-readable data are automatically translated into desired formats and routed into to the business logic of the *model*.

This separation of concerns between business logic and view is particularly useful in the case of WPF, because it enables designers and developers to use WPF's declarative XML-based interface markup language XAML to define the visual appearance and interactive behavior of a *view* without the need to know C# programming[30]. In the case of ZOIL, this allows designers to define the visual appearance of an object including complex interactive behavior and semantic zooming entirely in XAML (see details in section 4.3.5 and Figure 47) without having to write any UI code in C#. This is particularly useful in the light of present-day visual XAML editors with WYSIWYG functionality such as Microsoft Expression Blend and thus enables an improved designer/developer workflow.

3.4.5 Attached Behaviors Pattern

For designing object-oriented interaction, it is not sufficient to define an object's model and visual appearance alone. Objects also have to be assigned the desired interactive behaviors: Can an object be dragged, resized, or rotated? Does an object simulate physical behavior such as inertia or friction during these manipulations? Is the object a zoom target, so that a click or tap on it starts a zooming animation into the object until it covers the entire screen of the device?

To let designers easily assign such behaviors with XAML instead of C# code, the ZOIL framework makes extensive use of the *Attached Behaviors* architectural pattern[31]. The idea of *Attached Behaviors* is to define classes that contain the code for frequently used interactive behaviors and to attach them to an element of the user interface without having to change the element's class definition or creating and using subclasses. Thus *Attached Behaviors* can be used to make arbitrary user interface elements behave in a desired way while encapsulating the code for the actual behaviors into a class outside the elements' class hierarchy. This way, the ZOIL framework can provide designers with an extensible library of useful off-the-shelf interactive behaviors that can be easily assigned to user interface elements using XAML markup code. This facilitates iterative design without the need for writing procedural C# code or changing class hierarchies.

Figure 47 shows a XAML code sample that illustrates the use of *Attached Behaviors* for defining the interactive behavior of an object with ZOIL. The dummy object only consists of an icon that is defined in an external PNG image file ("DummyObjectIcon.png"). It becomes draggable, resizable, and rotatable with multi-touch and mouse by assigning the corresponding behaviors to the object in its XAML definition. Further behaviors of

[30] For space reasons, this thesis cannot provide an introduction into XAML. An overview of XAML for WPF can be found at *http://msdn.microsoft.com/en-us/library/ms752059.aspx* (Accessed Jul 21, 2012)

[31] The Attached Behavior pattern for WPF is introduced in a blog post by John Gossmann. *http://blogs.msdn.com/b/johngossman/archive/2008/05/07/the-attached-behavior-pattern.aspx* (Accessed Apr 27, 2012).

the ZOIL framework that control zooming and drag & drop are discussed in greater detail in (*Jetter, Gerken, Zöllner et al. 2010b*).

```
<ZComponent x:Class="DummyObject"
  ZObjectDragDropBehavior.IsDraggable="True"
  ZObjectResizeBehavior.IsResizable="True"
  ZObjectRotateBehavior.IsRotatable="True">
  <Border>
   <Border.Background>
    <ImageBrush ImageSource="DummyObjectIcon.png" Stretch="Fill"/>
   </Border.Background>
  </Border>
</ZComponent>
```

Figure 47 – A XAML code sample of a dummy object in a ZOIL user interface that uses *Attached Behaviors* to enable user manipulations.

An approach similar to *Attached Behaviors* was suggested in HCI research by Brad Myers in (*B. A. Myers 1989*). He presented a model that encapsulates interactive behaviors into a few "interactor" object types: Although graphics and input events may vary significantly between different objects or devices, there are always essential commonalities in their behavior. For example, when dragging an object with the mouse, stylus, or finger, the object always moves to the current XY-position of the user input and is dropped as soon as this user input stops. Myers suggested capturing these common behaviors in reusable "interactors" while still providing a high degree of customizability. Thus the *Attached Behaviors* pattern can be regarded as an implementation of Myers model to achieve more reuse and less redundancy of UI code. Furthermore, thanks to the declarative XAML language, assigning ZOIL behaviors becomes a very easy task.

3.4.6 Input Device and Peer to Peer Communication using OSC

The ZOIL framework provides simple ways to connect a ZOIL client application to multi-modal input libraries or other kinds of input device middleware. ZOIL applications can use the stateless UDP-based protocol Open Sound Control (OSC)[32] to connect to open source tools like the Squidy library[33] (*Jetter, König, and Reiterer 2009*; *W. König, Rädle, and Reiterer 2010*) that facilitate the integration of external post-WIMP devices such as Anoto pens, Nintendo Wiimote, or the TUIO[34] multi-touch protocol.

Using OSC has several advantages for prototyping: First, clients stay tolerant to unavailable devices or changing infrastructures during their lifetime, since no permanent connections have to be established. Second, UDP packets can be broadcasted to all clients in the subnet and therefore sending to specific IP addresses is unnecessary. Instead, packet destinations can be specified above the IP layer using OSC addresses. This

[32] OSC is a protocol for communication among computers, sound synthesizers, and other multimedia devices that is also used for communication with post-WIMP input devices, e.g., multi-touch tracking systems. More information at *http://opensoundcontrol.org/introduction-osc* (Accessed Jun 20, 2012).

[33] Video of using ZOIL and Squidy with Anoto and Wiimote: *http://www.youtube.com/watch?v=Np1ODKU48do* (Accessed Jul 21, 2012).

[34] TUIO is an open framework that defines a common protocol and API for tangible multitouch surfaces. Technically TUIO is based on OSC. More information at *http://www.tuio.org/* (Accessed Jun 20, 2012).

is desirable in early phases of prototyping and experimentation where devices are often added or removed.

As mentioned in chapter 2, there are also many situations in which the position and size of the currently visible region on one device should be transferred to another device. For example, in the *Media Seminar Room* users can move the tangible 'camera lens' on a tabletop to control the content that is displayed on a remote large vertical display. Another example is the use of a tablet PC in *DeskPiles* to send the currently visible object to another display for discussion, e.g., a tabletop. In these scenarios, clients have to (continuously) send and receive messages containing the parameters of the visible region, i.e., the location within the information landscape and the region's size, using peer to peer communication. The ZOIL software framework enables this by providing custom OSC commands for zooming and panning that each client can send or receive. If desired, incoming zooming & panning commands are automatically executed by the framework. The decision for using OSC instead of providing communication via a messaging object in the ZOIL server is based on the time delays that the update intervals for transparent persistence and synchronization would introduce and that could impair the feeling of a direct and natural remote control.

3.4.7 Related Work

The ZOIL software framework is partially based on concepts that were introduced by related work on metaphors, frameworks, and software architectures for interactive spaces or ubiquitous computing environments. This section compares and relates the ZOIL software framework and its underlying metaphors to this work.

For space reasons, this section focuses the comparison only on the most relevant work that is directly related to ZOIL's goals or approaches. For a more comprehensive overview of the state-of-the-art of multi-person-display ecosystems and multi-display environments, I recommend (*Terrenghi, Quigley, and Dix 2009*) and (*Nacenta, Gutwin, Aliakseyeu et al. 2009*). A survey of software infrastructures and frameworks for interactive spaces can be found in the section about "Intelligent Environment Systems" in (*Endres, Butz, and MacWilliams 2005*). The most relevant work from these overview articles and surveys, including some more recent work, is introduced below.

Application-Orientation vs. Object-Orientation vs. Instrumental Interaction

Multi-display tools and systems such as PointRight (*Johanson, Hutchins, Winograd et al. 2002*), Teamspace (*Shih, Crone, Fox et al. 2004*), ARIS (*Biehl and Bailey 2004*), Dynamo (*Izadi, Brignull, Rodden et al. 2003*), or IMPROMPTU (*Biehl, Baker, Bailey et al. 2008*) extend or expand current interface paradigms (e.g., applications, windows, menus, pointer). They rely on the traditional application-oriented WIMP interaction model of the desktop metaphor that, according to Nacenta, is not necessarily optimal for co-located collaboration in multi-display environments (*Nacenta 2008*). In contrast, Nacenta suggests working on a new breed of interfaces that can revolutionize collaboration in co-

located spaces and that depart significantly from current established interaction techniques and standard WIMP interfaces.

The ZOIL paradigm lays the foundation for such a new breed of interfaces that depart from traditional WIMP interaction models. ZOIL user interfaces are not based on applications and application windows but use a zoomable information landscape and an object-oriented user interface for a seamless interaction with different information types without application boundaries. ZOIL introduces an object-oriented interaction model in which objects become the first-level citizen of the user interface and are in the focus of the users' attention and manipulation instead of WIMP applications and their windows.

A further alternative to WIMP's application-orientation is Beaudouin-Lafon's *"instrumental interaction"* (*Beaudouin-Lafon 2000*). Similar to OOUIs, instrumental interaction deconstructs monolithic WIMP applications to dissolve application boundaries. Inspired by the way we use physical tools or instruments in the real world, Beaudouin-Lafon deconstructs application functionality into *domain objects* and *interaction instruments*. The domain objects of instrumental interaction are comparable to the domain objects used in OOUIs with one essential difference: Unlike in OOUIs, objects in instrumental interaction do not provide functions or methods directly to the user. Instead, the *interaction instruments* serve as mediators or two-way transducers between the user and the domain objects. For example, if a user scrolls a document with a scrollbar, the scrollbar is not part of the document object but is an instrument. The physical actions that the user executes on the scrollbar instrument are translated by the instrument into internal commands that are sent to the object. These commands affect the relevant object (e.g., scrolling the document) and feedback is provided to the instrument and the user (e.g., updating the view of the document and the thumb of the scrollbar). Therefore, unlike in an OOUI, there is no direct interaction of a user with domain objects via their views but interaction is always mediated by *instruments*.

In conclusion, OOUIs and instrumental interaction both pursue new interaction models that deconstruct monolithic applications. However, they differ to what extent this deconstruction takes place and how it becomes visible to the user:

1.) OOUIs dissolve application boundaries by "*turning the application inside-out"* (*Nielsen 1993: 59*). The workspace is populated with objects and the functionality to work with them is directly attached to them without visible applications or application windows. The different available functions or methods are presented to the user as parts of an object's view (e.g., after zooming in) or inside menus attached to the view (e.g., pop-up or context menus). The key advantage of OOUIs for users is the use of inheritance and polymorphism when modeling the class hierarchy of the user objects: *"interactions should be consistent across objects of the same class; where possible, operations should be polymorphic - applicable to different object types. This reduces the number of interaction behaviors and simplifies the interface"* (*Collins 1995*).

2.) Instrumental interaction goes a step further than OOUIs. Similar to real-world tools, the instruments are decoupled entirely from the domain objects to work with. *"Brushes*

are not bound to painting on a canvas, they can also be used to paint on the wall or on the hand of the painter" (*Klokmose and Beaudouin-Lafon 2009*). Instrumental interaction supports reusability of instruments by users, since the same instrument can be used with different objects. Objects should be manipulatable by an instrument in ways not necessarily anticipated by the object. Instruments should be applicable to objects when and where it makes sense, even if they were not designed to do so in the first place (*Klokmose and Beaudouin-Lafon 2009*).

Achieving such a great extent of reusability and flexibility is a great challenge for developers and software architects. For this reason, Klokmose and Beaudouin-Lafon introduce the VIGO (**V**iews, **I**nstruments, **G**overnors and **O**bjects) software architecture pattern for *"ubiquitous instrumental interaction"* in multi-surface environments (*Klokmose and Beaudouin-Lafon 2009*). Similar to the MVVM pattern in ZOIL, VIGO is also based on MVC and separates domain *objects* from their *views*. However, unlike in ZOIL, users do not directly manipulate the *view* and, by this, indirectly also the *object*. Instead, they manipulate objects through the interaction *instruments*. In order to manipulate an object, an instrument queries the *governors* attached to the object to validate its manipulations. *Governors* also observe object changes and can also manipulate objects to implement side effects (e.g., synchronization between objects, the rules of a board game). VIGO's support of the reusability of components for the developer is based on the flexibility provided by the dynamic management of governors attached to objects (*Klokmose and Beaudouin-Lafon 2009*).

In ZOIL, reusability for developers is based on an appropriate use of inheritance when modeling and implementing the class hierarchy of UI objects in C#. In cases, where inheritance cannot be used, ZOIL's *Attached Behaviors* pattern provides an alternative means to reuse implementations of interactive behaviors. In the sense of Bederson et al., the object-oriented interaction and its implementation in ZOIL is primarily *"monolithic"*. This means that it is primarily based on the use of concrete class hierarchies in which there is a strong mapping between software objects and real-world things. Furthermore, its implementation is primarily based on compile-time inheritance (*Bederson, Grosjean, and Meyer 2004*). Monolithic toolkits tend to have a few large classes containing the core functionality likely to be used that is extended using inheritance.

VIGO's instrumental interaction, however, is based on a clear separation of concerns among its strongly decoupled components. Bederson et al. would consider its design as *"polylithic"* (*Bederson, Grosjean, and Meyer 2004*), because its primarily based on run-time composition of many small classes with isolated bits of functionality, e.g., views, instruments, governors, objects. While VIGO's polylithic nature brings virtues such as a greater generality and flexibility (*Klokmose and Beaudouin-Lafon 2009*), this can also largely increase the complexity of using the architecture. Bederson et al. write about their experience with polylithic tookits: *"(...) the penalty is that the toolkit objects become more abstract, and the application writer must learn to work with a greater number of classes and with more relationships between objects. For certain classes of users, this may*

make the toolkit too hard to use. We have observed that significantly greater learning time is required to learn to use compositional toolkits. The resulting code quality may suffer, or users may be unable to discover how to implement a working solution at all. As an alternative, toolkit designers may use inheritance and hardwire design decisions into the class hierarchy. This yields a toolkit which, in our experience, is far easier to learn and to program against. The resulting code is also slightly more compact and easier to read. But, the penalty is that the policies adopted by the toolkit are harder to modify or repurpose" (*Bederson, Grosjean, and Meyer 2004*).

Bederson et al.'s observations strongly resonate with our findings during ZOIL's API usability evaluation that is discussed in section 3.5.7 (p.102): Participants had difficulties with ZOIL's use of the MVVM pattern and found it hard to use and feeling artificial. They would have preferred more concrete mappings between user interface objects and code objects. In the light of the rather simple separation of concerns in MVVM, VIGO's more abstract separation appears even more demanding and therefore might exclude certain classes of designers and developers, who conceptualize user interfaces rather in terms of their perceptible visual and interactive components than in terms of views, instruments, governors, and objects.

Metaphors for Structuring and Navigating Collaborative Workspaces

The ZOIL software framework differs from related work by distributing an object space across multiple clients to provide a virtual shared workspace in the form of a zoomable information landscape. The landscape can be accessed and explored using different interactive devices for zooming and panning and each device serves as a camera that provides an independent view of the landscape from above (see Figure 14 in section 2.1, p.33). This "device as a camera" metaphor touches three related kinds of concepts, approaches, or metaphors for structuring and navigating virtual workspaces:

1.) The first kind are collaborative *3D virtual worlds* with first person navigation in a workspace using mouse and keyboard, e.g., Alan Kay's Croquet (*Smith, Raab, Reed et al. 2004*), Second Life[35], or multi-player online games. Instead of first person navigation in such 3D virtual worlds, ZOIL's planar zoomable user interface intentionally reduces the 5 or 6 degrees of freedom of 3D navigation to lower the users' cognitive load during view control and spatial orientation. ZOIL's information landscape is always viewed directly from above and panning does not change the angle but only the position of the virtual camera. Thus panning only happens along two axes and users only have to zoom in or out. Furthermore, semantic zooming ideally reduces the effort for accessing desired data and functions by providing the appropriate size-dependent representation. More details on the history of zoomable user interfaces and the role they play for ZOIL are provided in chapter 4.

[35] Second Life is a commercial 3D virtual world and online community for chat and online meetings: *http://www.secondlife.com* (Accessed Jul 22, 2012).

2.) The second kind of concepts use *augmented reality* or create *mixed-reality rooms*. They try to reduce the cognitive load of navigating in virtual information and functionality by employing the real-world physical environment as a referential domain. For example, Fitzmaurice's Chamaeleon uses spatially aware palmtop computers to let users view, browse, and manipulate a virtual 3D space (e.g., a 3D tree hierarchy of data) by physically moving the palmtop display in the real world. Thus, to the user, the palmtop appears to be a see-through lens or camera bridging computer-synthesized information and physical objects (*Fitzmaurice 1993*). A similar approach with the metaphor of a *"peephole"* is used by Yee (*Yee 2003*). Yee uses mobile position-tracked handheld PDAs to provide a window on larger 2D and 3D virtual workspaces, e.g., large maps or 3D visualizations.

Butz et al.'s EMMIE system uses see-through head-worn displays for augmented reality that overlay a 3D virtual environment on the physical environment of a meeting room (*Butz, Höllerer, Beshers et al. 1999*). This provides a collaborative augmented environment as a *"hybrid user interface"* in a shared physical space that combines virtual elements, such as 3D widgets, and physical objects, such as tracked displays and input devices. In subsequent work, Butz & Krüger use several peepholes in an instrumented environment with combinations of fixed and steerable projectors, large displays, and portable devices to create mixed-reality rooms (*Butz and Krüger 2006*). They demonstrate several small applications in which physical and virtual objects coexist in the room and can be accessed, searched, manipulated, and organized by the users' navigation, movement, and actions in physical space, e.g., moving or turning to different displays or pieces of furniture, wiping virtual objects between displays, attaching virtual annotations to physical objects or locations.

Unlike these approaches, ZOIL does not make use of position-tracking to create spatially aware devices yet[36]. In ZOIL's current form, physical movement of devices or users does not affect the visible region of the information landscape on a device but is only used to simply carry devices around. Instead, users use explicit interactions such as multi-touch gestures or pen input, or they move tangible user interface elements to control a device's visible region and to manage the objects in the information landscape.

3.) The third kind of concepts does not use space to organize content or functionality. They neither use virtual space, e.g., 3D worlds or zoomable user interfaces, nor physical space, e.g., moving and seeing through a spatially aware mobile device as a peephole, as the referential domain. The referential domain is *"non-spatial" (Nacenta, Gutwin, Aliakseyeu et al. 2009)*.

In its simplest form, this can be realized using a shared file system or a cloud storage service such as DropBox[37] to access shared content and functionality from multiple

[36] As discussed in section 1.3.2, other researchers (e.g. Roman Rädle) of the HCI Group at Konstanz are currently investigating how to combine spatially aware proxemic interactions with the ZOIL approach.

[37] DropBox is a popular cloud storage server for file hosting: *http://www.dropbox.com* (Accessed Jul 22, 2012).

different devices. All objects can be accessed by using the file path, file name, or URL. Although, it is common to talk about "storage locations" or "locations in a file system", they must be considered non-spatial from a cognitive point of view.

IMPROMPTU is an interaction framework that allows users to better realize natural and effective group work practices when working in MDEs (*Biehl, Baker, Bailey et al. 2008*). IMPROMPTU enables users to share all kinds of application windows with other users by placing them on the other users' personal display, e.g., a desktop or laptop PC, or on public large displays. The interface for performing these actions shows the names and photos of collaborators, so that the non-spatial social environment becomes the referential domain.

Streitz et al.'s i-LAND uses physical passage-objects, e.g., a key chain, a watch, or a pen, that are acting as physical "bookmarks" into the virtual world (*Streitz, Tandler, Müller-Tomfelde et al. 2001*). By putting a passage-object on an identification sensor that is integrated into interactive tabletops or interactive walls, a virtual "bridge" appears on the device. This bridge reveals all virtual objects from the shared workspace that are "bookmarked" by the object. Users can move objects out of the bridge on to the device to work with them or they can move objects from the device into the bridge to "bookmark" them. Thus the non-spatial referential domain is the identity of objects.

A non-spatial referential domain can be easily implemented with ZOIL, simply by mapping a non-spatial attribute to the location of a region in the information landscape. Thereby the non-spatial attribute (e.g. a person's name) serves as a "bookmark" into the information landscape. For example, the information landscape can contain a region that contains multiple rectangular subregions. Each of these subregions might contain further subregions or it is used as a container for storing virtual objects inside of them. The above-mentioned non-spatial examples can be easily realized by mapping a file path, a person's name, or a passage-object's identity on the location of one of the different regions or its subregions to "bookmark" them. Similar as in i-LAND, putting a physical object on a device running a ZOIL application would then lead to a jump zoom into this region to make its content accessible. In the same way, entering a person's name or entering a hierarchical file path could be used to access desired region.

Software Infrastructures and Frameworks for Interactive Spaces

The main source of inspiration for ZOIL is Streitz et al.'s pioneering vision of i-LAND and its *"roomware"* (*Streitz, Geißler, Holmer et al. 1999*; *Streitz, Tandler, Müller-Tomfelde et al. 2001*). They introduce example scenarios, roomware devices, e.g., interactive walls, chairs, tabletops, and also a software framework in Smalltalk. Similar to ZOIL, Streitz et al.'s COAST framework provides a shared object space across all devices and a virtual location metaphor for structuring collaboration. However, as mentioned above, i-LAND uses a non-spatial metaphor, while ZOIL uses the virtual zoomable information landscape for navigating and organizing digital objects in continuous space and scale. Unlike i-LAND, ZOIL thus enables smooth navigation and seamless transitions between virtual locations and digital objects (see also chapter 5).

Similar to the vision of i-LAND, the iRoom and its iROS meta-operating system enable collaborative use of one or more large displays with portable devices, e.g., laptops, in a local physical space (*Johanson, Fox, and Winograd 2002*). One key contribution of iRoom are new GUI widgets and interaction techniques for fluid interaction with a pen device on large displays, e.g., the FlowMenu. The iRoom also provides components such as PointRight (*Johanson, Hutchins, Winograd et al. 2002*) for pointer redirection that allows a pointing device on any laptop in a room to serve as a pointer on any other laptop or large public display. iRoom applications such as Teamspace (*Shih, Crone, Fox et al. 2004*) enable users to "push" any file or URL from one machine onto the display of another by opening the file there in the window of a locally installed application. Alternatives to the GUI's traditional application-oriented desktop or file system are not part of this research. In contrast, ZOIL focuses on rethinking the established WIMP paradigm and achieves a seamless interaction without visible application boundaries using zoomable and object-oriented user interfaces.

The iRoom's underlying meta-operating system iROS provides the *Event Heap* as a core subsystem. iROS applications do not communicate directly with one another; instead they use indirection through the *Event Heap*. It stores and forwards messages and provides a central repository to which all applications in the room can post events. An application can selectively access events on the basis of pattern matching fields and values (*Johanson, Fox, and Winograd 2002*). Thereby iROS aims at a minimal barrier-to-entry for developers, a goal that was also shared during ZOIL's development (see following section 3.5). However, unlike iROS, ZOIL or also Shared Substance (see below) provide a much closer connection between data and events which results in a conceptually simpler and more coherent approach for the developer (*Gjerlufsen, Klokmose, Eagan et al. 2011*).

Similar to iROS, the Gaia meta-operating system is a distributed middleware infrastructure that coordinates software entities and heterogeneous networked devices contained in a physical space (*Román, Hess, Cerqueira et al. 2002*). Gaia's functionality is much broader in scope than iROS or ZOIL and offers five basic services: 1.) An *event manager* service that distributes events in the active space. 2.) A *context service* that applications can use to query and register context providers such as sensors that track people's locations or room conditions like temperature and sound. 3.) A *presence service* maintains updated information about present resources and detects the presence of devices, people, or software entities. 4.) The *space repository* stores information about all software and hardware entities in the space and lets applications browse and retrieve entities on the basis of specific attributes. It uses the *presence service* to learn about entities entering and leaving the space. When an application starts executing, it uses the space repository to find appropriate resources such as execution nodes, displays, and speakers. This lets developers describe applications generically and map them to the available resources in different environments. 5.) The *context file system*, is a context-aware file system that uses a virtual directory hierarchy to retrieve data not only from hierarchies of folders but also from virtual folders that contain the results of simple

queries. Files can be accessed by their storage location but also by an associated context, e.g., a person present in the space or a room temperature.

In conclusion, Gaia has a very different focus than the ZOIL software framework. It is focused on dynamic resource detection and allocation, distribution of execution, and context-aware querying of files. It provides the functionality of an operating system to implement ubiquitous computing applications. However, it does not provide any dedicated support for implementing post-WIMP interaction or novel visualization techniques in the applications which is the focus of ZOIL. Therefore, GAIA's and ZOIL's functionality should rather be considered complementing and not competing.

Hugin is a Java-based framework for information visualization that, similar to ZOIL, supports real-time co-located and remote collaboration using multiple tabletops (*Kim, Javed, Williams et al. 2010*). Hugin's features for visualizations and access control are more advanced than those integrated in ZOIL. However, the size of Hugin's shared workspace is currently restricted to the physical size of the screen. Thus, unlike in ZOIL, zooming and panning cannot be used to create multiple regions, for example for supporting mixed-focus collaboration and fluid transitions between coupling styles in space and scale (see chapter 5).

Shared Substance is a recent data-oriented framework and distributed application model for multi-surface environments (*Gjerlufsen, Klokmose, Eagan et al. 2011*). Its goal is to create a technical foundation that allows users to share both physical and digital resources and interact with them freely. Each client application or device uses the nodes of a shared tree structure to store or access the application state, system resources, or input device streams. These nodes can be selectively shared with, replicated on, and mounted on other clients using discovery mechanisms without a central server. Similar to VIGO (*Klokmose and Beaudouin-Lafon 2009*), interaction in Shared Substance is not based on object-oriented but instrumental interaction (*Beaudouin-Lafon 2000*). Therefore, unlike in ZOIL, data and behavior are separated and loosely coupled using a data-oriented polylithic design based on runtime-composition and a great a degree of abstraction. As discussed above, this can lead to a greater complexity when developers want to use the framework. In comparison, ZOIL's monolithic design based on compile-time inheritance is less flexible but easier to use for implementing ZOIL's object-oriented interaction. Furthermore, unlike ZOIL's shared information landscape, Shared Substance does not provide persistency for the shared tree structure.

3.5 API Usability Evaluation of the ZOIL Framework

The following sections introduce the results of two empirical studies of using the ZOIL software framework for OOUI implementation with student designers and developers. These API usability evaluation studies were conducted to learn how successful the ZOIL software framework is in providing support to developers of OOUIs for post-WIMP interactive spaces. For this purpose, the studies focused on the two key qualities of user interface software tools that were identified by Myers et al. (*B. Myers, Hudson, and*

Pausch 2000): the *threshold* ("how difficult it is to learn how to use the tool?") and the *ceiling* ("how much can be done using the tool?").

According to Myers et al., the ideal tool has a *low threshold* and a *high ceiling*. This means that it is easy to get started with basic functionality (*low threshold*) and that there is a gentle slope of the learning curve that ultimately leads to a great sophistication of what can be created (*high ceiling*). Furthermore, there should be few *walls*. This means that designers should not be forced to often stop and to learn something entirely new to reach a greater level of sophistication. Therefore, the purpose of our studies was to focus on the *threshold* and *ceiling* of the ZOIL software framework and to identify typical *walls* during its use.

To achieve this, the goal of the studies was to collect rich qualitative data from realistic project settings, instead of measuring purely quantitative data (e.g., task times or task completion) in artificial controlled experiments. For this reason, a new data gathering method for API usability evaluation was created and applied (*Gerken, Jetter, Zöllner et al. 2011*). In addition, participants were also asked for subjective assessments of threshold and ceiling using questionnaires. While subjective assessments are of course not a reliable measure, they still can be considered a rough indicator for ZOIL's API usability.

3.5.1 Concept Maps for API Usability Evaluation

Data gathering methods for API usability evaluation are an ongoing research topic in HCI. As discussed in (*Gerken, Jetter, Zöllner et al. 2011*; *Gerken 2011*), API usability evaluation is particularly challenging: Using established HCI evaluation methods, e.g., a direct observation of an API user's interaction with the GUI of an IDE during given tasks, does not reveal the details of the interaction with the API itself and thus is vulnerable to subjective interpretation. Furthermore, finding ecologically valid tasks for the limited timeframe of an observation session is difficult or even impossible for complex APIs like the ZOIL software framework. Using a complex API involves a learning process over days if not weeks. Thus traditional observation techniques are prone to discover only how usability problems surface during lab sessions without achieving a deeper understanding of the actual learning barriers and when and where the mental model of the API user conflicts with the API's intended use. Simple retrospective interviews with API users are also problematic, as they lack artifacts that trigger a sufficiently detailed and efficient discussion with the participant.

In (*Gerken, Jetter, Zöllner et al. 2011*), we[38] therefore suggested and applied a novel data gathering method based on *concept maps*. Concepts maps were originally introduced by

[38] The concept maps method was conceived of by Jens Gerken and is the main contribution of his PhD thesis on longitudinal research in HCI *(Gerken 2011)* and our joint CHI publication (*Gerken, Jetter, Zöllner et al. 2011*). I assisted Jens Gerken in adapting the concept maps method for API usability evaluation by extending the visual notation of the concept maps with directed named links to better describe functional relations between API components. Furthermore, I provided the ZOIL software framework as the API to study. Therefore I supported him during the analysis and interpretation of the study's findings on ZOIL's API usability to enable assessing the method's practical value for discovering relevant API design flaws. The practical application of the method during our study was also assisted by Michael Zöllner, who helped with organizing study materials, capturing data, and interpreting participants' maps.

Novak (*Novak and Gowin 1984*) in the late 1960s and early 1970s as a research method to assess children's understanding of science concepts. They can be described as visual knowledge representations with nodes and edges. Each node represents a concept and is linked with one or several other nodes via edges. The edges are typically directed and labeled to describe the nature of the connection between two nodes.

Figure 48 – A concept map created by student designers and developers during a study of ZOIL's API usability. Source: (*Gerken, Jetter, Zöllner et al. 2011*)

Our proposed concept mapping approach is based on asking participants to visualize the relationship between the concepts of the application to develop and the concepts of the used software framework or API (Figure 48). Depending on the nature of the study, the participants can be handed an initial set of concepts of the framework that they can use as a starting point to enable better comparisons between the results of different participants. The actual mapping exercise is done during a 30 to 60 min observation session, which is video-taped and recorded using a thinking-aloud protocol. For each

participating group, such a session is repeated (e.g. once a week during a five week period) depending on the complexity of the application that has to be implemented. During these repetitions, the users do not start from scratch but only alter their concept map from the previous session. They are asked to change only those parts of their concept map that they do not perceive as correct representations of their mental model of the API anymore. How their mental model has changed is thus implicitly reflected in those changes.

By analyzing the resulting maps from each session, the creators of the framework are enabled to understand typical misconceptions, usability problems, and the learning process over several sessions. The data gathered in the concept maps and the thinking-aloud protocols can reveal where the mental models of the creators and the users of the framework deviate. This enables a deeper understanding of the reasons for API usability problems and the identification of walls and learning barriers. Further details about this evaluation method can be found in (*Gerken, Jetter, Zöllner et al. 2011*).

3.5.2 Study 1 – Hotel Browser

The first study was conducted as a 5-week follow-up activity to the study of OOUI design and modeling described in section 3.2 and used the same teams of participants and the same assignment of creating a ZOIL-based hotel browser application. However, during this second part of the study, the participants received further functional requirements for the prototype that they should implement within the additional 5 weeks. The extended functionality included the full set of features from Figure 42 (p.64) without rotation and resizing but including the additional requirements of filtering hotels by different user criteria. Furthermore, a shopping cart for selecting favorite hotels was introduced (see Appendix 9.3). During this follow-up study, a total of five weekly concept mapping sessions were conducted for each team. By comparing the participants' maps to our own map of the ZOIL framework (see Appendix 9.3), we gained insight into the ZOIL software framework's threshold, ceiling, and walls during the ongoing development of the prototypes.

An example for a resulting prototype is shown Figure 49. The top left screen shows how hotels are visualized on a zoomable map and different controls on the top are available for filtering hotels by the hotel stars, the hotel features, and the desired range for the price per night. The top right screen shows how textual comments or additional images from an URL can be attached to an existing hotel using a drag-and-drop operation that creates a visual link between them. For accessing individual hotels in a cluster on the map, users can zoom into the cluster that is then shown in a rectangular layout using a semantic zooming approach (bottom left). Individual hotel details and associated user-generated content such as photos can be accessed by further zooming in (bottom right).

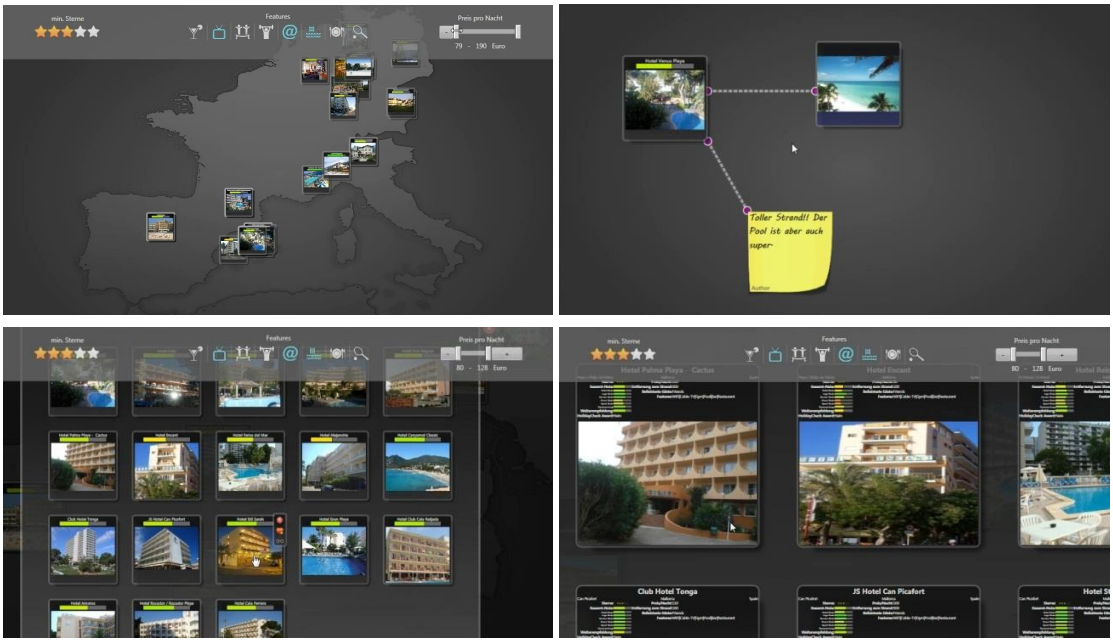


Figure 49 – One of the resulting *Hotel Browser* prototypes from study 1 created by students Marcio Alvarez di Castro and Till Niese.

3.5.3 Study 2 – Self-chosen Projects

In a second study, 8 participants (graduate level students of computer science and mathematics) were working in teams of 2 on individual self-chosen projects. The study lasted 4 weeks beginning with an introductory two hour lecture about the ZOIL framework. After that, each of the 4 groups worked independently. During 3 interview sessions (one per week), each group created and updated a concept map to reveal their mental model of the ZOIL framework. Additionally, each group used an online diary in a Wiki to report on encountered problems and their solution. The resulting prototypes are described in the following.

MeSearch

The first group created the *MeSearch* prototype for searching scientific publications from the ACM Digital Library on a Microsoft Surface tabletop (Figure 50). It uses a SQL database containing an export of the digital library to search in the metadata of all publications. After entering a keyword into a search field, ZOIL's information landscape is populated with objects that represent the found documents. Users can establish own spatial arrangements of documents using multi-touch gestures. Semantic zooming is used to gradually reveal all details of a publication, e.g., full title, authors, abstract, and also functionality for printing or mailing. By further zooming into the author names, the authors' email address and a function for sending an email is revealed. Two additional regions appear on the top left ("Cited By") and bottom right ("References") of a publication. By zooming into the "Cited by" region, a new information landscape appears that contains the publications that are citing the current publication. By zooming into the "References" region, a new information landscape appears that contains the publications that are cited by the current publication.

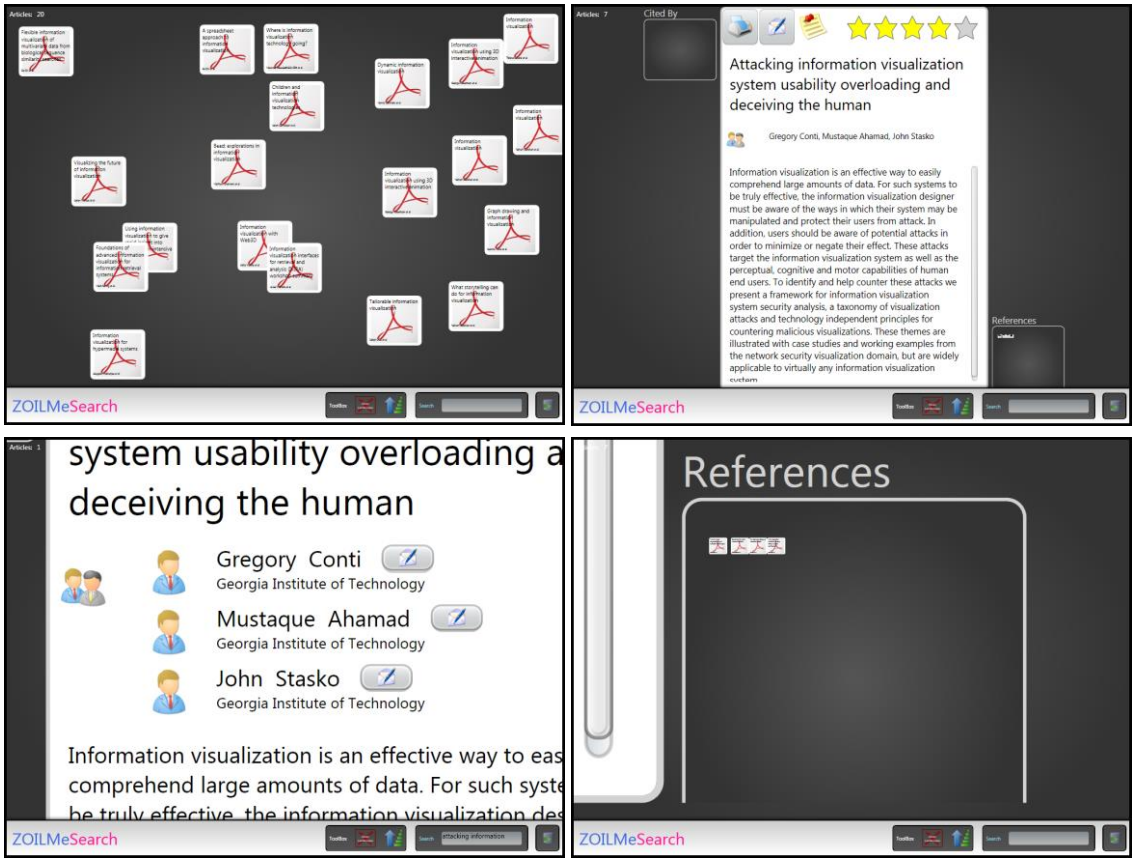


Figure 50 – *MeSearch* is a tabletop system for searching and exploring the ACM Digital Library.

ZOIL Activity Manager

The *ZOIL Activity Manager* of the second group uses ZOIL's information landscape to provide users with a visual meta-layer above the file system for spatially managing information items, e.g., files like excel sheets, mails from Microsoft Outlook, URLs to web pages, together with user-generated notes (Figure 51). It is deployed as an application for Windows-based devices and thus can be used on desktop PCs, tablet PCs, or pen-enabled large vertical screens. Using the Windows clipboard or drag-and-drop, the ZOIL landscape in the application window can be populated with information items (Figure 51: 1. & 2.). Semantic zooming into items is used to reveal a preview or an editor control and buttons for opening the associated application (Figure 51: 3. & 4.). Users can organize items in space and scale to create clusters of items and notes related to a user's activity. By drawing lines between items, visual links can be established which can be clicked or tapped for accelerated navigation between connected items using automated zooming and panning animations (Figure 51: 5. & 6.). This project inspired what later became *DeskPiles* (see section 2.2).

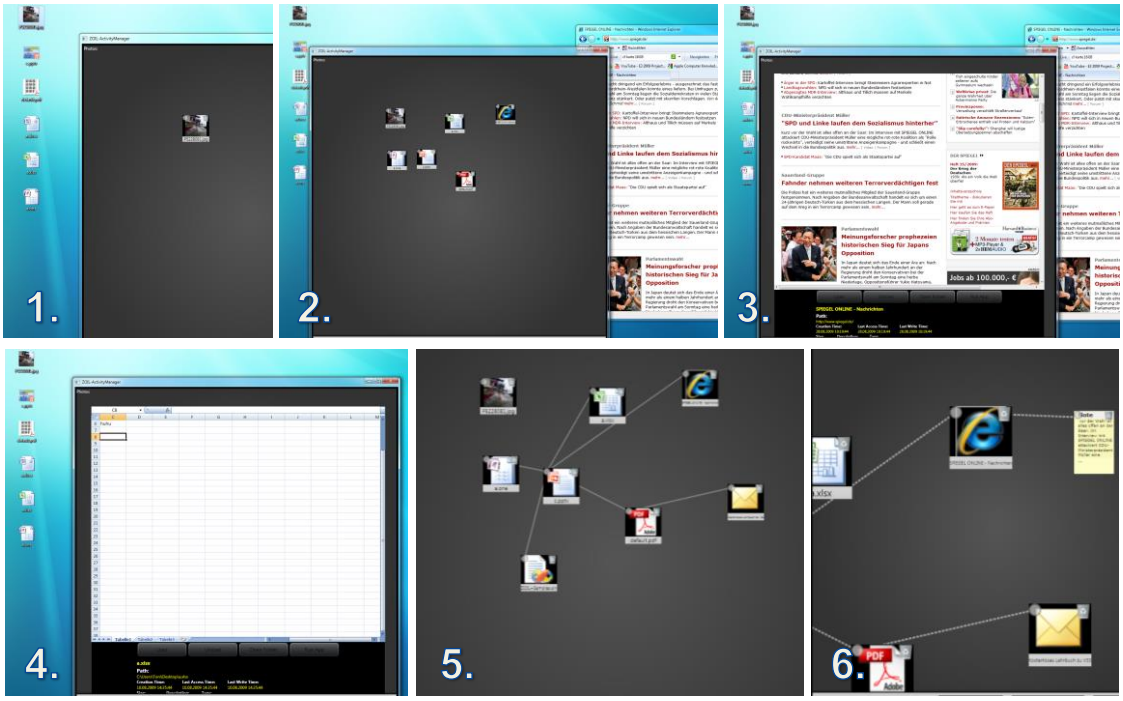


Figure 51 - The *ZOIL Activity Manager* enables users to organize files, notes, and bookmarks in ZOIL's information landscape. It serves as a meta-layer above the file system for activity management.

Gone fishing... Movies

The *Gone fishing... Movies* prototype of the third group extends the multi-user and multi-device *Media Seminar Room* (see section 2.1) by integrating a playful search and filtering interface for movies (Figure 52 left) on the Microsoft Surface tabletop. It is displayed as an overlay in front of ZOIL's information landscape. Its size can be switched between four sizes ranging from a minimized view in the bottom left corner to a maximized view covering the entire screen (Figure 53).

Different filter and search criteria can be specified for movie metadata from the IMDb[39]. For example, after touching user-generated tags floating in the fish tank to "catch" them, movies relevant to these tags are enlarged in the information landscape and get more opaque. Unrelated movies shrink and get more transparent. It is also possible to select tags from the entire set of user-generated tags in the IMDb using a dial control for accelerated access (Figure 52 right). Similarly, quantitative dynamic range queries, e.g., "production year between 1970 and 1980", are possible using range sliders (Figure 54) and similarly affect the size and opacity of movies.

[39] Internet Movie Database. *http://www.imdb.com* (Accessed Jul 24, 2012)

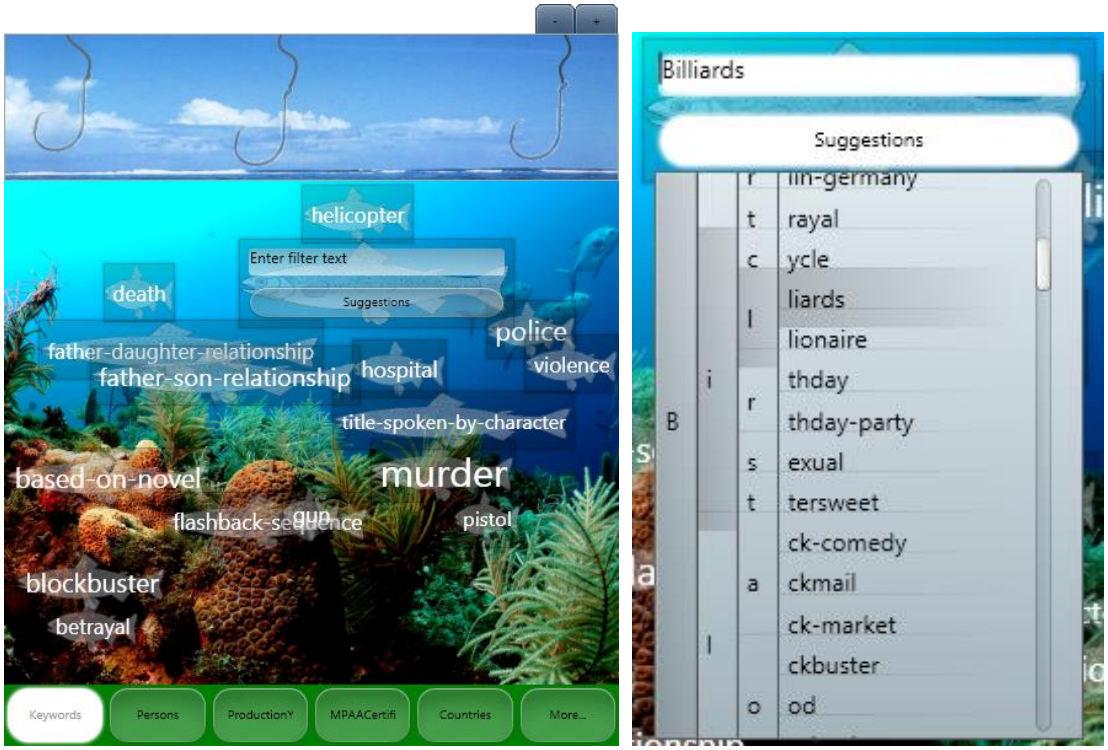


Figure 52 – The *Gone fishing … Movies* prototype is a playful search interface for movies that is integrated into the *Media Seminar Room*. User-generated tags float in a fish tank and can be "catched" by touching them (left). Tags can also be selected from large lists with a dial control (right).

Figure 53 – The size of the filter overlay can be switched between four sizes.

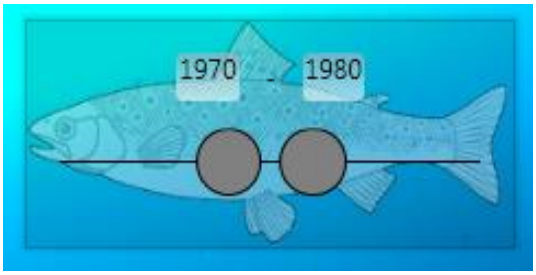


Figure 54 – A range slider for the production year of movies.

Visualization Dashboard (VizDash)

The *VizDash* prototype of the fourth group uses ZOIL's information landscape on the Microsoft Surface tabletop to present a visualization dashboard with different business charts. The charts are generated by querying a SQL database containing business data and are rendered using controls from the Visifire[40] library that features different kinds of line charts, bar charts, and pie charts with animated transitions. The purpose of the dashboard is to provide an instant overview of the current state of an enterprise. Details of a chart can be accessed on the tabletop by zooming in. Particular interesting charts can be sent from the tabletop to one of the two large vertical displays for group discussion and comparison.

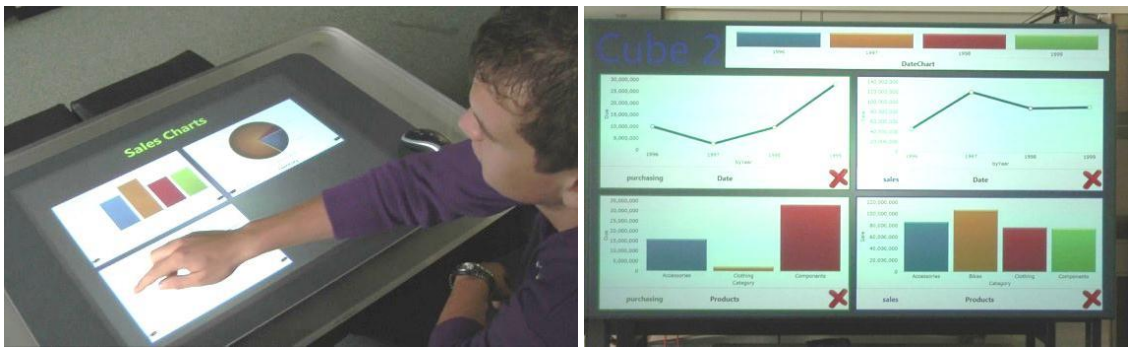


Figure 55 – *VizDash* uses a tabletop (left) and two large vertical high-resolution displays (right) for the collaborative generation, exploration, and presentation of charts from a business database.

3.5.4 ZOIL in Research Practice

Before discussing the findings from the two API usability studies in terms of ZOIL's ceiling and threshold, this section introduces a third source of empirical data. As mentioned in section 1.6.2, the ZOIL software framework has been used by other researchers from our group to study novel user interfaces in the context of their research on interactive spaces, Reality-Based Interaction, and collaborative interfaces for search or creative design, e.g., (*Geyer and Reiterer 2010*; *Geyer, Pfeil, Budzinski et al. 2011*; *Geyer, Pfeil, Höchtl et al. 2011*; *Geyer, Budzinski, and Reiterer 2012*; *Heilig, Demarmels, Rexhausen et al. 2009*; *Heilig, Demarmels, Allmendinger et al. 2010*; *Heilig, Huber, Gerken et al. 2011*; *Heilig 2012*; *Jenabi 2011*).

[40] Visifire for WPF. *http://www.visifire.com* (Accessed Jul 24, 2012).

While these prototypes have not been created during an observational study following a clearly outlined procedure, they still can serve as a source of information about what can be achieved when other researchers are using the framework and thus about ZOIL's ceiling. Therefore, selected examples are presented here to enrich the discussion of study findings in the following sections.

MedioVis 2.0

MedioVis 2.0 is a "Knowledge Media Workbench" (*Heilig, Demarmels, Rexhausen et al. 2009*) that employs ZOIL design principles and an early version of the ZOIL software framework to provide library users with a visual interface to a digital movie library on a large vertical high-resolution screen or a Microsoft Surface tabletop (Figure 56).

Figure 56 – MedioVis 2.0 is a single-device "Knowledge Media Workbench" that can be used on a large vertical screen or on a tabletop in a library. Source: (*Heilig, Demarmels, Rexhausen et al. 2009*).

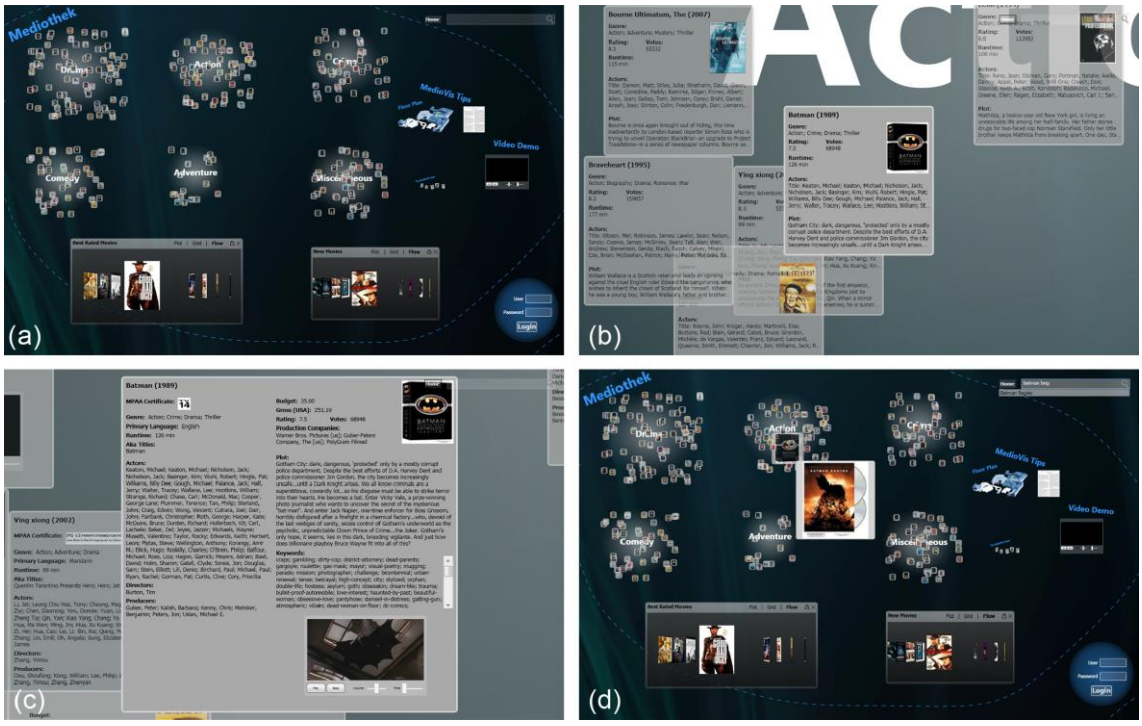


Figure 57 – (a) Overview of all movies clustered by genre. (b+c) Semantic zooming into a cluster and into a movie. (d) keyword search enlarges relevant items. Source: (*Heilig, Demarmels, Rexhausen et al. 2009*).

Thereby *MedioVis 2.0* makes use of semantic zooming to zoom from the information landscape with an overview of all movies (Figure 57a) into the details of individual movies (Figure 57b,c). Search is possible by entering a keyword. This results in a growth of relevant movies in the information landscape by their relevance ranking (Figure 57d). As discussed in section 4.3.6, the system also makes intense use of see-through visualization lenses such as scatter plots or cover flows (see Figure 90, page 133).

MedioVis 2.0 is a local single-device application and was implemented before the ZOIL software framework contained a client-server architecture for multi-device interactive spaces. Later, the *Media Seminar Room* (see section 2.1) used many concepts of *MedioVis 2.0* and introduced them into a multi-device context.

Search Token

The *Search Token* is a tangible tabletop user interface for collaborative search. It was first published in (*Demarmels, Huber, and Heilig 2010*) and (*Heilig, Demarmels, Allmendinger et al. 2010*) and enables users to specify multiple search terms for querying a movie database by putting physical search tokens on the tabletop and attaching a search term using on-screen keyboards (Figure 58). The defining feature of the search tokens is that they are combined using weighted Boolean logic and users control the weight of each token by rotating it clockwise or counterclockwise. Assigning a weight of 1.0 or greater to a search term expresses that this search term is desired. Movies containing this search term are increased in size by multiplying their current size by a weight greater than 1.0. Assigning a weight between 0.0 and 1.0 expresses that this search term is not desired. Movies containing this search term shrink size by multiplying their current size by a weight smaller than 1.0. The content of the database is visualized in ZOIL's information landscape underneath the search tokens. The position of objects can be manually modified by users using multi-touch manipulations and the sizes of objects are constantly updated depending on the state of the query formulated with the search tokens.

In (*Heilig, Huber, Gerken et al. 2011*), Heilig et al. compared user strategies, working styles, and patterns of communication during around-the-table collaboration using *Search Token* with those during collaboration using synchronized single-user interfaces (Figure 59). For realizing the second experimental condition in Figure 59 (right), they used the ZOIL client-server architecture to synchronize the state of three search applications using ZOIL's information landscape as a shared workspace. The design, implementation, and user study are also described in greater detail in the Master's thesis of Mischa Demarmels (*Demarmels 2010*).

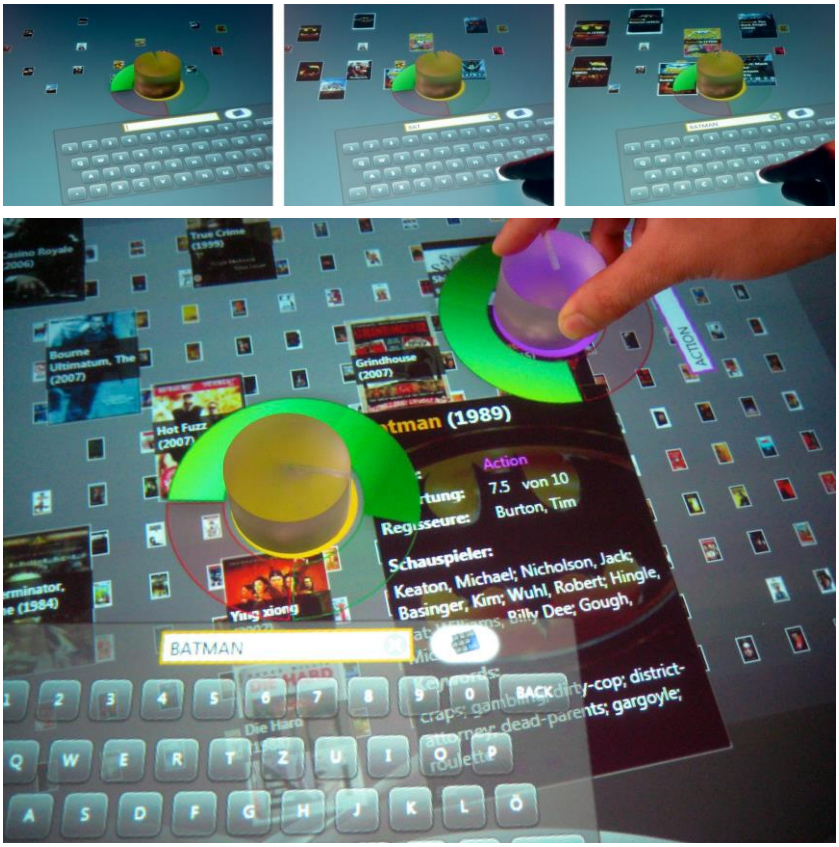


Figure 58 – The *Search Token* tabletop system for collaborative search based on weighted Boolean queries. The search term assigned to a token is editable in a text entry field. The weight assigned to a token is visualized with a colored circular indicator. Source: (*Heilig, Huber, Gerken et al. 2011*).

Figure 59 – Around-the-table collaboration with *Search Token* (left). Second experimental condition with synchronized single-user non-tangible user interfaces (right). Source: (*Heilig, Huber, Gerken et al. 2011*).

AffinityTable

The *AffinityTable* is a hybrid surface for supporting collaborative design activities (*Geyer, Pfeil, Budzinski et al. 2011*; *Geyer, Pfeil, Höchtl et al. 2011*). It combines digital pen & paper with an interactive tabletop and tangible tokens to support affinity diagramming. An additional vertical 4K high-resolution display is used to support reflection and group coordination (Figure 60).

Figure 60 – The *AffinityTable* consists of a tabletop as shared action space (a) and a vertical 4K high-resolution display as shared reflection space (b). Source: (*Geyer, Pfeil, Höchtl et al. 2011*).

AffinityTable lets users create and arrange digital artifacts for affinity diagramming by writing with digital pens on small paper notes and putting them on the tabletop. By putting a note onto the tabletop, the note and its content is identified and a digital representation appears at its location in ZOIL's information landscape that serves as *AffinityTable's* shared workspace. After populating the workspace with notes, users can use different tangible and multi-touch interaction techniques to arrange, align, and cluster digital artifacts and to zoom and pan in the workspace for reflection and discussion. Thereby the table is primarily used for direct interaction with digital artifacts (*shared action space*, Figure 60a), while the vertical display is primarily used for reflection (*shared reflection space*, Figure 60b). However, since both devices are ZOIL clients, they provide independent views of ZOIL's shared zoomable information landscape and it is possible to flexibly change their roles during collaboration.

Figure 61 – *IdeaVis* provides a hybrid workspace and interactive visualization for paper-based collaborative sketching sessions. Sketches are created on paper (center) and are collected and arranged on a vertical 4K high-resolution display (left). A touch-enabled interactive visualization is provided to enable creative facilitators to explore, examine, and support the success of a session (right).

IdeaVis

A further ZOIL-based system for supporting creative design processes is *IdeaVis (Budzinski 2012; Geyer, Budzinski, and Reiterer 2012)*. A ZOIL information landscape is displayed on a vertical 4k high-resolution display and is populated with paper-based

sketches during a collaborative sketching session (Figure 61). By capturing when and by whom sketches are printed, altered, reprinted, or copied, the system collects enough data to visualize the course of a session in real-time on an additional facilitator display. Instead of using ZOIL's standard visualization of the information landscape, the facilitator display visualizes the shared object space as a hyperbolic tree with the different sketches as nodes and the course and branches of sketching as edges. Additional functionality for managing the sketches and session are provided after semantic zooming into the nodes.

For space reasons, this section does not describe further examples of the use of the ZOIL software framework in other's work, e.g., (*Fäh 2011*; *Jenabi 2011*; *Runge 2010*). However, Figure 62 in the following section provides an overview of all work that has been created with the ZOIL software framework to this day and discusses it to illustrate ZOIL's ceiling.

3.5.5 ZOIL's Ceiling

The resulting prototypes from the two API usability studies, the examples from chapter 2, and the prototypes from the work of other researchers illustrate the many different usage scenarios and different levels of complexity and maturity that can be achieved with using the ZOIL software framework. A complete overview of all prototypes created with the framework, including some prototypes such as *EuroITV* (see section 4.3.2 and Figure 76, page 123) or experimental setups (see section 4.5 and Figure 96, page 143) that are introduced in later parts of this thesis, is provided in Figure 62. The overview in Figure 62 uses following dimensions to summarize the different examples:

1. *Nature* of the example: Is it a *mature prototype* (MP) supporting real-world tasks that was used and evaluated with users? Is it an *experimental setup* (ES) that supports only a very focused task for the purpose of controlled experiments? Is it a *technology demonstrator* (TD) that illustrates selected design principles or features of ZOIL, but is not intended for user studies? Is it a result of one of the two API usability *evaluation* (EV) studies with student designers and developers?
2. *Domain*: Short description of application domain and scenario of use.
3. *Publications*: List of the publications mentioning or based on this example.
4. *Figures*: List of the figures of the example that can be found in this thesis.

The overview in Figure 62 contributes to the assessment of the *ceiling* of the ZOIL software framework ("how much can be done using the tool?"). The ZOIL software framework has been used for creating prototypes for very different domains such as search, sensemaking, information management, creative design, and cross-display interaction techniques. To this day, two PhD theses (not including this thesis), six Diploma or Master's theses, and two Bachelor's theses are based on prototypes using the ZOIL software framework. A further PhD thesis by Florian Geyer which strongly builds upon ZOIL is going to be finished this year. Furthermore, two journal publications refer to prototypes built with ZOIL (*Elmqvist, Vande Moere, Jetter et al. 2011*; *Jetter, Zöllner, Gerken et al. 2012*). Apart from this, there are three posters or workshop and demo

papers based on prototypes with ZOIL. Most importantly, there are nine peer-reviewed conference papers using ZOIL-based prototypes from which two were awarded with honorable mention awards at renowned scientific conferences such as ACM Creativity & Cognition (*Geyer, Pfeil, Höchtl et al. 2011*), and the ACM SIGCHI Conference on Human Factors in Computing Systems (*Jetter, Gerken, Zöllner et al. 2011*).

In conclusion, the ZOIL software framework can be considered an appropriate tool for prototyping in the context of HCI research. It can be used to implement focused single-user, single-device, experimental setups for controlled experimentation such as the *Apparatus: Mouse vs. Multi-Touch* or much more sophisticaed multi-user and multi-device interactive spaces that are evaluated with user groups during realistic user sessions with realistic task such as *AffinityTable*, *IdeaVis*, or *Facet-Streams*.

A further source of information about ZOIL's *ceiling* is to look at the qualitative and quantitative data collected during the two API usability evaluations. At the end of the first study, on a scale from 1 (very low ceiling) to 5 (very high ceiling) the mean rating of the 11 participants was 3.6 (SD = 0.5). At the end of the second study, on a scale from 1 (very low ceiling) to 7 (very high ceiling) all 4 participating groups have assessed ZOIL's ceiling with 5. While these ratings are above neutral, they hint at room for improvement. The interviews and concept maps during the studies revealed a series of shortcomings of the current state of the ZOIL framework that negatively affected the assessments.

Hosting Win32 Controls in WPF

Since ZOIL is based on WPF, it also inherits a critical weakness of this platform. Currently there are no native WPF controls available for rendering or editing popular content such as Web pages, PDF documents, Flash applications, or Microsoft Office documents (e.g., Word, Powerpoint, Excel, or OneNote). Although existing Win32 COM or ActiveX controls can be hosted inside a WPF application, they do not support WPF's scaling and rotation transformations and thus are not usable in ZOIL's zoomable information landscape or with multi-touch rotation or resizing on tabletops. This is due to WPF's "airspace problem": Hosted controls are not rendered by WPF itself but always render themselves on top WPF content. Thus they are agnostic of WPF's underlying rendering transformations such as scaling or rotation.

A workaround to the scaling problem can be achieved by manually initiating updates on hosted controls during zooming animations, but this approach is very costly and increases CPU load dramatically. Another approach is to use a native WPF third party control for rendering Web pages. For example, Awesomium[41] uses Google's Chromium Web rendering engine for native support of Web content in WPF applications. However, this approach is also very resource-intensive and during the studies users were discouraged to use it by the additional overhead and failed attempts to host PDF or Office Web browser plugins.

[41] Awesomium is a windowless Web-Browser framework. *http://www.awesomium.com* (Accessed Jul 25, 2012).

Example	Nature[42]	Domain, Scenario of Use	Publications	Figures in this thesis
AffinityTable	MP	Creative Design: Support for collaborative affinity diagramming	(*Geyer, Pfeil, Budzinski et al. 2011*; *Geyer, Pfeil, Höchtl et al. 2011*)	Figure 60 (p.96), Figure 94 (p.136)
Apparatus: Mouse vs. Multi-Touch	ES	ZUI Navigation: Experiment for studying differences of input devices	(*Jetter, Leifert, Gerken et al. 2012*; *Schubert 2010*)	Figure 96 (p.143), Figure 97 (p.145), Figure 99 (p.148)
DeskPiles	MP	Information management: Creation, presentation, and discussion of results of lab work in nanophotonics	this thesis and *DeskPiles* technical report[43]	Figure 5 (p. 6), Figure 22 - Figure 29 (p.38-43), Figure 81 (p.127)
Distributed Sketching	TD	Creative design: Collaborative sketching and management of design-related artifacts	(*Geyer, Jetter, Pfeil et al. 2010*)	Figure 30 - Figure 33 (p.44-45)
EuroITV	TD	Information management: personal information management of photos, email, movies, and notes with an interactive television	(*Engl 2008*; *Jetter, Engl, Schubert et al. 2008*)	Figure 76 - Figure 78 (p.123-p.125), Figure 80 (p.127)
Facet-Browsing	TD	Search: Faceted navigation by zooming into multiple layers of a movie database	(*Heilig 2012*; *Runge 2010*)	none
Facet-Streams	MP	Search: Collaborative faceted product search on a tabletop using a hybrid visual-tangible surface	(*Elmqvist, Vande Moere, Jetter et al. 2011*; *Jetter, Gerken, Zöllner et al. 2010a*; *Jetter, Gerken, Zöllner et al. 2011*; *Zöllner 2012*)	Figure 5 (p.6), Figure 34 - Figure 38 (p.46-50)
Gone fishing … Movies	EV	Search: Playful search & filtering user interface as extension to the *Media Seminar Room*	this thesis	Figure 52 - Figure 54 (p.91-92)
Hotel-Browser	EV	Sensemaking: Filtering, spatial organization, and annotation of search results from a hotel catalog incl. user-generated content	(*Jetter, Zöllner, Gerken et al. 2012*)	Figure 49 (p.88)

[42] MP: Mature Prototype, ES: Experimental setup, TD: Technology demonstrator, EV: Results from evaluation study.

[43] *http://research.microsoft.com/en-us/projects/beneath-the-surface/bts_cambridge_metasurfacing_with_the_surface_finalreport.pdf* (Accessed Oct 9, 2012)

IdeaVis	MP	Creative Design: Support of creative design according to the "Brainsketching" method for collaborative sketching	(*Budzinski 2012*; *Geyer, Budzinski, and Reiterer 2012*)	Figure 61 (p.96)
Media Seminar Room	TD	Sensemaking and annotation of search results from a movie catalog	this thesis	Figure 10 - Figure 20 (p.31-37)
MedioVis 2.0	TD	Search: Visualization, filtering, and spatial organization of search results from a movie catalog	(*Heilig, Demarmels, Rexhausen et al. 2009*)	Figure 56 (p.93), Figure 57 (p.93)
MeSearch	EV	Search: Searching & browsing in ACM Digital Library.	this thesis	Figure 50 (p.89)
MultiDragger	ES	Cross-display Interaction: Enables movement of objects between ZOIL clients using bimanual pointing techniques in mid-air.	(*Fäh 2011*)	none
PrIME	ES	Cross-display Interaction: Enables pointing, selection, movement, and storage of objects in a multi-display environment with a mobile phone	(*Jenabi 2011*)	none
Search Token	MP	Search: Collaborative tangible search using multiple search tokens for weighted Boolean queries	(*Demarmels, Huber, and Heilig 2010*; *Heilig, Demarmels, Allmendinger et al. 2010*; *Heilig, Huber, Gerken et al. 2011*; *Heilig 2012*)	Figure 58 (p.95), Figure 59 (p.95)
VizDash	EV	Sensemaking: Exploration and visualization of business data using a visualization dashboard with business charts.	(*Jetter, Zöllner, Gerken et al. 2012*)	Figure 55 (p.92)
ZOIL Activity Manager	EV	Information management: Activity-centered spatial management of files, notes, and other information items on a visual meta-layer above the file system.	this thesis	Figure 51 (p.90)

Figure 62 – Complete overview of all prototypes created with the ZOIL framework.

In the *DeskPiles* prototype, this problem was avoided by creating static views of content by converting pages or documents to PNG or XPS formats which are natively supported in WPF. However, this resulted in inconsistencies after annotating static views in the information landscape because these annotations only exist inside the ZOIL application and cannot be written back into the original pages or documents.

A satisfying solution to this problem cannot be achieved in the ZOIL framework alone but must happen inside the WPF framework. A solution to the "airspace problem" was announced for future versions of WPF[44], but it still remains unsolved in all the official WPF releases available from Microsoft.

ZOIL's Drag-and-drop Support

Participants in study 2 were unsatisfied with ZOIL's support of drag-and-drop. While making objects change their location with touch or mouse was easy to achieve with ZOIL's *Attached Behaviors*, participants asked for an equivalent support of drag-and-drop for initiating actions, e.g., after dropping objects on objects. Although this problem can be easily solved by implementing own behaviors on the application level, the effort for this was apparently perceived as too great and thus negatively influenced ZOIL's perceived ceiling during the evaluation. In future, an improved support for drag-and-drop in ZOIL could be achieved by using an approach similar to the Silverlight Toolkit extension for Microsoft Silverlight. This toolkit introduces a series of new *Attached Behaviors* for defining drag-and-drop behavior using XAML that could also be implemented for WPF inside ZOIL.

Nested Information Landscapes

During the development of the *MeSearch* project, the first group in study 2 reported problems when trying to nest multiple information landscapes. Input events in the nested information landscape were not correctly transformed in its coordinate system leading to unpredictable behavior when dragging objects. In the meantime, according code changes to the ZOIL framework have been made to solve this problem.

Touch-support for Non-Touch Controls

In some cases participants made use of controls on the tabletop that do not support touch input. For example, in the *VizDash* prototype the controls for business charts from the Visifire library were not touch-enabled at the time and thus could only be used on the Surface when connecting a mouse. Since Windows 7 is now generally touch-enabled and the new Surface tabletop (Samsung SUR40 with Microsoft PixelSense) provides a mouse-emulation for touch events, this problem does not exist anymore. Nevertheless, at the time of the studies, this problem still existed and participants demanded for a solution for this in the ZOIL software framework.

[44] WPF vNext: Microsoft will solve the 'Airspace problem'. *http://bartwullems.blogspot.de/2010/11/wpf-vnext-microsoft-will-solve-airspace.html* (Accessed Jul 25, 2012).

Limitations of ZOIL's ZUI Implementation

Finally, in some cases, participants reported problems with the overall rendering performance of WPF and the ZOIL framework. In scenarios with many hundreds or thousands of complex objects in the information landscape, the frame rate drops and system response time increases noticeably. In some cases this leads to a barely usable user interface. A more detailed discussion of these limitations is provided in section 4.4.

Summary

Despite these shortcomings of the current state of the ZOIL framework, the diversity of prototypes created during the study (e.g., single vs. multi-user, single vs. multi-display, integration of ZOIL UI into WIMP environment) reveal a high ceiling of the ZOIL framework. In particular, the framework benefits from its use of C#, .NET and WPF because of its interoperability with most APIs and SDKs, e.g., Microsoft Surface SDK or 3rd party libraries such as Visifire. This is also true for different data sources, e.g., SQL databases for the Windows operating system.

3.5.6 ZOIL's Threshold

Due to its nature as a software framework that builds upon a complex API such as WPF, the ZOIL software framework cannot be considered a low-threshold toolkit or authoring system for rapid prototyping. Even when using visual XAML editors such as Microsoft Expression Blend and making use of *Attached Behaviors*, all projects lasted 4 or 5 weeks and they involved development effort in C# and the more verbose XAML language with the lines of code varying between 700 (*VizDash*) and 2.800 (*ZOIL Activity Manager*).

To get a better estimate for the necessary development effort, we asked the participants of study 1 to self-report the hours they spent on the project. The result was a mean of 15 hours per week and per participant, so 1/3 to 1/2 of the working time of a fulltime developer. While this clearly indicates a high threshold, it is still encouraging regarding the fact that half of the participants in study 2 and all participants in study 1 had no prior experience with C#, WPF, and XAML at all. This explains the slightly below neutral assessments of ZOIL's threshold by the 4 groups of study 2 with a mean score of 3.25 (SD = 0.96) on a scale from 1 to 7 (1: easy to learn, 7: difficult to learn) and the 11 C# and WPF novices of study 1 with a mean score of 2.6 (SD = 0.92) on a scale from 1 to 5 (1: easy to learn, 5: difficult to learn).

Based on these observations, ZOIL's threshold can be considered low enough that a 2 person team of C# and WPF novices is able to create a multi-device post-WIMP UI with ZOIL in a fulltime project of 2-3 weeks.

3.5.7 ZOIL's Walls

Typically, the first wall that participants encountered was to understand the role that the object-oriented database db4o plays for the ZOIL software framework. In ZOIL, the role of this database is to provide a client-server architecture for sharing the object space with ZOIL's information landscape across device boundaries. ZOIL uses this database to

achieve persistence and synchronization and thus also enables independent views of a shared visual workspace (see sections 3.4.1 and 3.4.3).

This role was often confused with the more traditional role that relational databases play for Web applications. For example, the group working on *VizDash* initially tried to import the relational business data for their business charts as table objects into db4o. Only after we intervened to clarify the role of db4o, the group realized that for their application scenario it is more suitable to use a standard relational database, e.g., MS SQL server, for the business data and keeping the role of db4o focused on persistence and synchronization between the table top and the vertical screens.

Another frequently mentioned wall is the *Model-View-ViewModel* pattern. Since understanding this pattern needs previous knowledge about WPF concepts such as data binding and commands, participants did not consider it as self-explaining, even when code samples were present. Only one group of the nine groups participating in study 1 and 2 considered the *Model-View-ViewModel* pattern as easy to understand and to apply.

The interviews and concepts maps also revealed a reoccurring theme in connection with the *Model-View-ViewModel* pattern that is also relevant for post-WIMP implementation of OOUIs in general: All participants strongly approved of *Attached Behaviors*. Although the underlying mechanisms were mostly not understood in detail, the simple way of selecting and assigning behaviors to objects in XAML was considered as “logical” or “natural”. On the contrary, groups in both studies criticized the *Model-View-ViewModel* as overly complicated, unnecessary, and feeling artificial compared to the naturalness of *Attached Behaviors*.

This hints at a more general theme that is analogous to the common desire for concrete mappings and monolithic toolkits in OOP (*Bederson, Grosjean, and Meyer 2004*). Similar to the user of a post-WIMP OOUI who perceives an object as a single physical entity (e.g., a virtual or tangible object on a tabletop with size, mass, friction), post-WIMP programmers would like to create such objects and define their behaviors and properties without creating many abstract artificial software objects that have no counterparts in the real world. While *Attached Behaviors* support this kind of concrete mapping and natural programming, the *Model-View-ViewModel* pattern does the exact opposite and disintegrates a UI object into three components from which only the *view* is directly perceptible and has physical, real-world dimensions. While participants in the study accepted the necessity for a separation of *view* and *model*, they expressed that they still would strongly prefer more natural mappings between code objects and real-world objects and more declarative approaches for defining properties and behaviors. These findings are particularly relevant for the discussion of alternative post-WIMP patterns or frameworks such as VIGO or Shared Substance in section 3.4.7 which have an equally abstract or even less concrete mapping than *Model-View-ViewModel*.

4 Zoomable User Interfaces

This chapter introduces *Zoomable User Interfaces* (ZUIs) and discusses their key role within the ZOIL paradigm. ZOIL uses ZUI principles to replace traditional concepts of the WIMP desktop metaphor such as files, folders, and application windows. Instead, ZOIL introduces a spatially continuous zoomable workspace (the 'information landscape') as a model-world UI for natural and consistent management of digital information items and interaction with virtual tools.

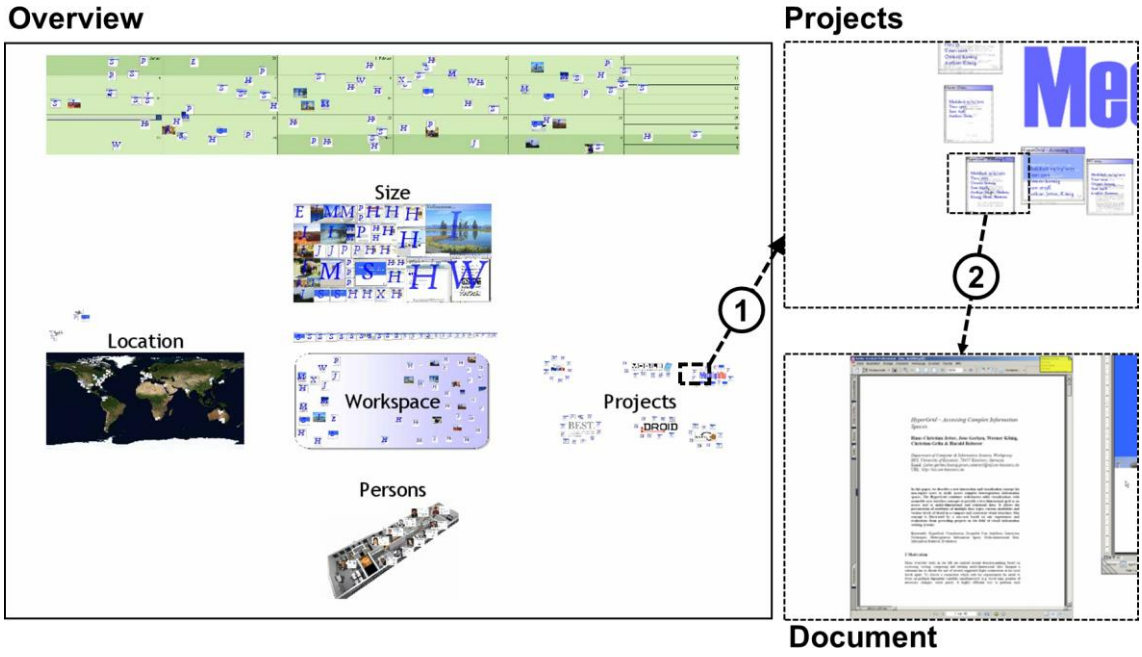


Figure 63 – An early vision of ZOIL for document management from (*Jetter, König, Gerken et al. 2008*): The 'information landscape' features different metadata dimensions or facets as entry points into the document space (e.g. location, file size, projects, persons). Users can smoothly zoom from an overview into the facets (1) and eventually into the documents themselves (2).

In a first step, this chapter discusses ZUI history and the foundations and mathematics of ZUI interaction. Then it presents ZUI examples taken from different ZOIL-based prototypes. It continues with a user study of the effect of multi-touch interaction on ZUI navigation and spatial memory. Where appropriate, this chapter also gives an overview about the implementation of ZUIs in and with the ZOIL software framework. It concludes with the formulation of the 2^{nd} *ZOIL design principle*.

This chapter contains ideas, results, figures, and text from my previous journal, conference, and online publications: It shows my designs of *EuroITV* from a demo paper on ZUIs for interactive television (*Jetter, Engl, Schubert et al. 2008*) and from the *DeskPiles* project with Jeremy Baumberg of the NanoPhotonics Centre of the University

of Cambridge and Natasa Milic-Frayling of Microsoft Research Cambridge. It uses paragraphs and sections about the advantages of ZUIs for ZOIL from two journal publications (*Gerken, Heilig, Jetter et al. 2009*; *Jetter, Zöllner, Gerken et al. 2012*), a workshop paper (*Jetter, König, Gerken et al. 2008*), and a full paper at HCSE 2010 (*Jetter, Gerken, Zöllner et al. 2010b*). The sections about ZUI implementation in the ZOIL software framework use figures, paragraphs, and code samples originally created for (*Jetter, Zöllner, and Reiterer 2011*; *Jetter, Zöllner, Gerken et al. 2012*; *Zöllner, Jetter, and Reiterer 2011*). This chapter also reproduces text, figures, and results from our user study on multi-touch vs. mouse navigation that was published at AVI 2012 in (*Jetter, Leifert, Gerken et al. 2012*).

4.1 ZUI History

This and the following section provide a brief overview of ZUI history and their mathematical foundations, so that readers can better follow the designs, experimental results, and recommendations presented in this chapter. Apart from this chapter, the authoritative review of ZUIs in (*Cockburn, Karlson, and Bederson 2008*) is recommended for further reading.

4.1.1 Visual Space as User Interface

The history of ZUIs in academic research started with the seminal “Spatial Data Management System” or SDMS (Figure 64). It was created in 1976 by the MIT’s Architecture Machine Group that later became the MIT Media Lab. SDMS revolutionized the user interface by providing a virtual zoomable 2D plane for spatially managing data instead of using a text-based command or programming language or a stack of many overlapping application windows on a single screen. This plane was called “Dataland” and was displayed on a large projector screen in a special-purpose “Media Room” that also contained smaller touch-enabled monitors and joysticks for input (*Bolt 1984*; *Donelson 1978*). Thus SDMS can be regarded as the first prototype of a multi-display environment and post-WIMP interactive space.

Figure 64 – The “Media Room” of the MIT running the ‘Spatial Data Management System’ on a large projector screen and two smaller touch-enabled monitors. Source: (*Bolt 1984*).

The planar “Dataland” contained heterogeneous multimedia data such as virtual letters, books, maps, or videos. It also provided office tools such as a phone or calculator. The user could navigate to the locations of these different objects by panning the plane with a joystick and zooming into the object of interest. While the large projection provided a blown-up image of this object, a smaller “world-view” monitor always showed an overview of the entire plane to keep the user from getting lost (Figure 65). This overview+detail technique predates even those in 1980’s video games that Cockburn et al. assumed to be the first examples of this kind (*Cockburn, Karlson, and Bederson 2008*).

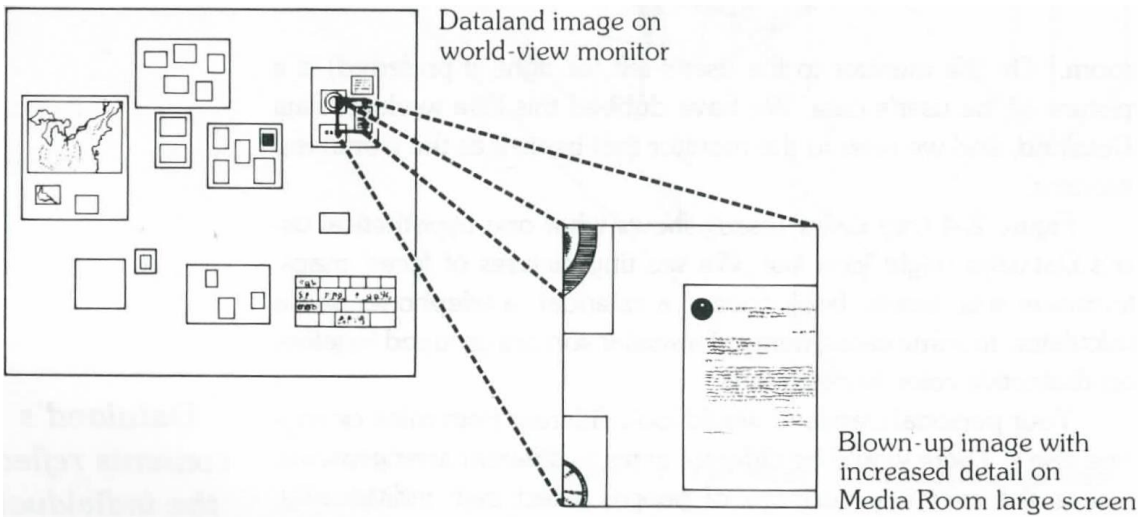


Figure 65 – Overview and detail in SDMS with smaller world-view monitors and a large screen. Source: (*Bolt 1984*).

The zoom-in operation gradually revealed multiple discrete layers of increased visual detail and functionality. At a maximum level of detail, the tools like a calculator or the content of a book became interactive and could be operated on one of the smaller touch-enabled monitors without overlaying application windows (Figure 66).

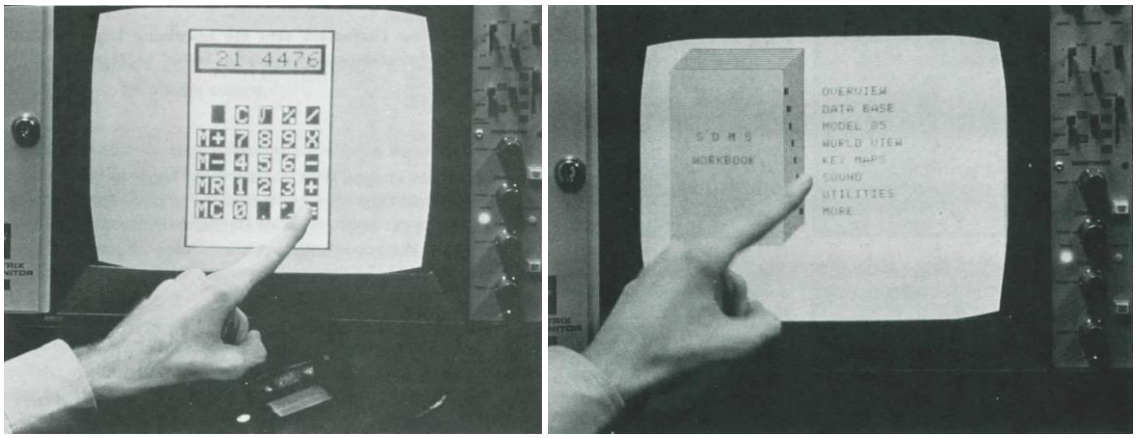


Figure 66 – Zooming into SDMS's calculator tool and the digital book. From: (*Bolt 1984*).

Dataland did not serve as a visual browser to an underlying file system like the WIMP desktop metaphor, but instead it itself actually contained all the necessary data and tools. Bolt summarized this design philosophy as *“Dataland isn’t a map of the data; it is*

the data" (*Bolt 1984: 12*). By this, Dataland introduced an early notion of object-oriented instead of application-oriented interaction. It also replaced the hierarchical file structures and file names of the mainframe age with purely visual organization schemes. This resonates with the vision of interaction designer and GUI pioneer Jef Raskin who two decades later similarly suggested to organize information in a zoomable plane (*Raskin 2000: 149*). Instead of a file system with folders and file names, he suggested to organize information in spatial structures and layouts that are not anymore a part of the interface, but a part of the content.

4.1.2 From Dataland to Data Mountain

The idea for the use of space in SDMS and Dataland is based on suggestions of psychologist George A. Miller (*Bolt 1984: 5*). In 1968, he hypothesized that computer data might be better managed on a spatial basis and he suggested to create computers that make use of their users' spatial memory and cognition to represent information (*Miller 1968*). Furthermore, the creators of SDMS also mention the lyric poet Simonides of Ceos and his "Method of Loci" from ancient Greece as an important source of inspiration (*Bolt 1984: 3*). This "Method of Loci" is a mnemonic technique also known as "memory palace" and is based on a mental walk in an imaginary building. By memorizing objects or events in connection with imaginary locations, it exploits the eidetic and spatial disposition of human memory (*Wagner 2000*). In a similar fashion, Jef Raskin also argued for using a zoomable plane to create a more natural and less *"mazelike"* user interface: *"We can find things (...) because we tend to remember landmarks and relative position, a fact sometimes called the psi effect and long known to psychological researchers. (...) you go right to a particular document because you remember that it is just to the left of the orange piece of paper that Aviva put up."* (*Raskin 2000: 153*). However, Raskin does not point to literature from psychology that scientifically explores this "psi effect" or experimentally validates his claim.

While visuo-spatial representations of computer data became quickly very popular, it took HCI researchers until 1998 to use quantitative experimentation for examining to what extent visual representations of space on the screen can really exploit users' spatial memory and cognition. For example, in the "Data Mountain" study (*Robertson, Czerwinski, Larson et al. 1998*), a perspective representation of a plane that contained miniature Web pages proved to be an effective alternative to traditional folder hierarchies of Web favorites (Figure 67). The storage times in users' memory were increased and the users' retrieval times and retrieval failures were reduced. However, in 'Data Mountain' the perspective and the visual frame of reference always remained constant. Neither zooming & panning navigation nor post-WIMP input devices were a part of the experiment, leaving a core question about ZOIL unanswered: How much can users really benefit from a Dataland-like ZUI in post-WIMP environments?

Subsequent work introduced the 3D window manager "Task Gallery" (*Robertson, Dantzich, Robbins et al. 2000*). Here tasks were placed as artwork on the walls of a virtual art gallery (Figure 67). In a user study, users could successfully recall a great majority of

the tasks they had placed, so that the authors concluded that 3D approaches for window management are promising. However, they also reported that the metaphor compromised the ways in which space was used. For example, users had not used the available space on the ceiling and floor as they had believed that this would violate the gallery metaphor. *"Some participants simply did not like the idea of tasks lying on the floor"* (*Robertson, Dantzich, Robbins et al. 2000*). As described below, this resonates with the approach of *"informational physics"* of Bederson et al. (*Bederson, Hollan, Perlin et al. 1996*) who intentionally avoided overly concrete metaphors and used metaphor-free and less realistic representations of space.

Figure 67 – Top: In the *Data Mountain* users arranged and memorized their bookmarks inside of a perspective visual representation. Source: (*Robertson, Czerwinski, Larson et al. 1998*).
Bottom: *Task Gallery* extended this concept to create a 3D virtual gallery as window manager. Source: (*Robertson, Dantzich, Robbins et al. 2000*).

4.1.3 Zoomable User Interfaces (ZUI)

The terms “multi-scale user interface”, “zoomable user interface”, or “zooming user interface” were introduced into HCI in the two seminal publications about “Pad” and “Pad++” by Perlin & Fox and Bederson et al. (*Bederson, Hollan, Perlin et al. 1996*; *Perlin and Fox 1993*). Both articles refer to SDMS as a source of inspiration, but provide a much more elaborated interaction model and a software implementation with smooth and continuous zooming (at least for simple visual elements and UI controls) for conventional desktop PCs instead of SDMS’s room-sized special purpose hardware and multiple screens.

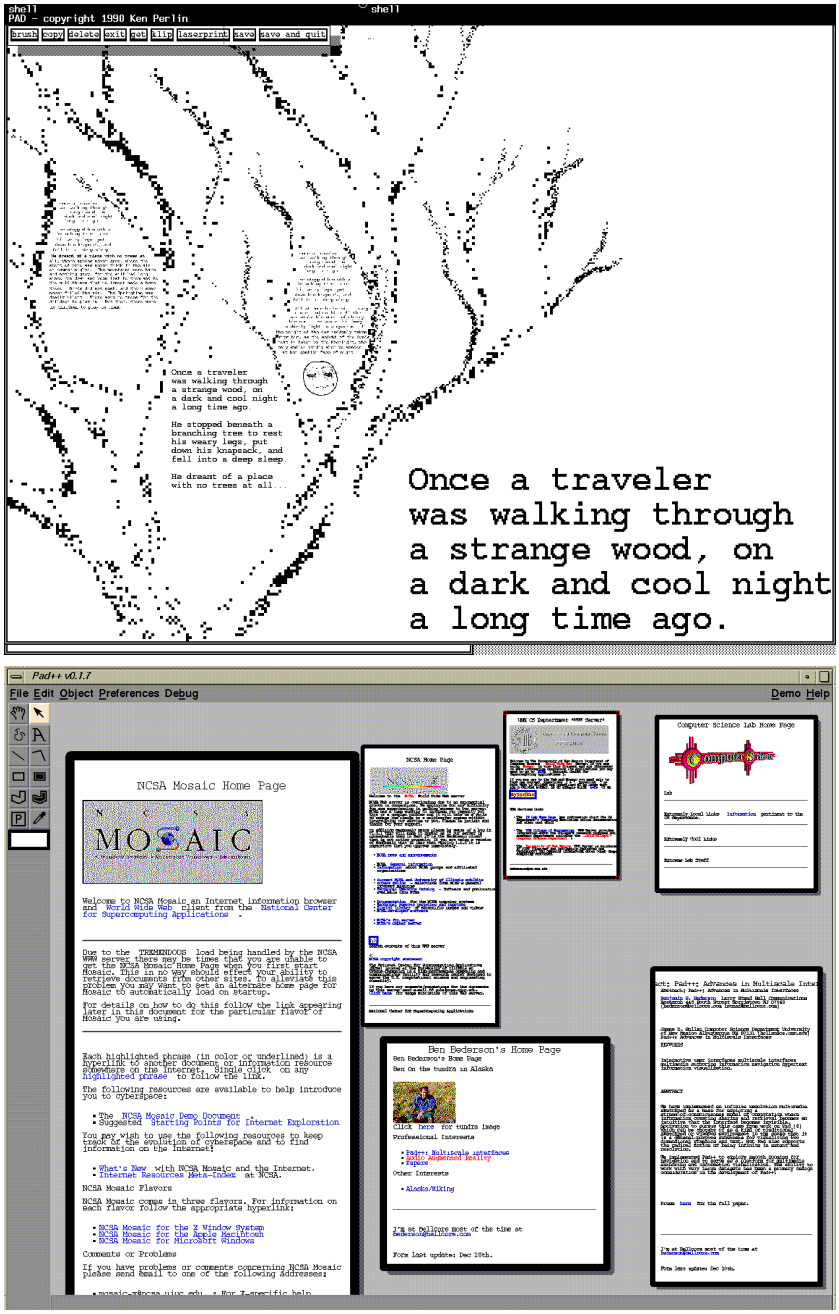


Figure 68 – Top: ‘Branching tree story’ realized with *Pad*. As the reader zooms into different branches of iconic text representations, different stories unfold. Source: (*Perlin and Fox 1993*).
Bottom: A zoomable Web browser realized with *Pad*++. Source: (*Bederson, Hollan, Perlin et al. 1996*).

Pad and Pad++ were a part of Bederson et al.'s larger strategy of using *"informational physics"* for user interface design. This strategy was intended as an alternative to the many metaphor-based design approaches of the time that often mimicked the mechanisms of earlier media and underutilized the possibilities of new digital media. In contrast, *"informational physics"* aimed not at finding familiar metaphors, but at *"the development of a physics of appearance and behavior for collections of informational objects. For example, an effective informational physics might arrange for an object's representation to be a natural by-product of normal activity"* (*Bederson, Hollan, Perlin et al. 1996*). By this, Bederson et al. wanted *"to start to more fully exploit radical new computer-based mechanisms"*. This design strategy is similar to post-WIMP designs of Reality-Based Interaction (see section 1.3.2, p.8) that exploit the users' familiarity with fundamental concepts and mechanisms from real-world experience (e.g. space, first law of Newtonian physics, persistence of objects) without using overly concrete real-world metaphors such as "THE OPERATING SYSTEM IS AN OFFICE DESKTOP" (*Imaz and Benyon 2007: 52*). In particular, ZUIs aim at facilitating user orientation and navigation in information spaces by *"tapping into our natural spatial and geographic ways of thinking"* (*Perlin and Fox 1993*). This emphasis on users' visuo-spatial abilities also resonates with the *"environment skills and awareness"* theme of Reality-Based Interaction (*Jacob, Girouard, Hirshfield et al. 2008*) or the power of Tim Rohrer's physiologically embodied metaphors (e.g. animated zooming windows) discussed in (*Imaz and Benyon 2007: 6*).

Bederson et al. describe ZUIs and the zooming & panning navigation in a large canvas using an analogy from the physical world: The computer screen resembles a sheet of a *"miraculous new material that is stretchable like rubber but continues to display a crisp computer image, no matter what the sheet's size"* (*Bederson, Hollan, Perlin et al. 1996*). Since this material can stretch orders of magnitude more than rubber, vast quantities of information can be organized at different places and sizes on the sheet. *"Everything you do on the computer is on this sheet. To access a piece of information you just stretch to the right part and there it is. (...) The beginnings of an interface like this sheet exists today in a program we call Pad++. We don't really stretch a huge rubber-like sheet, but we simulate it by zooming into the data"* (*Bederson, Hollan, Perlin et al. 1996*).

Apart from zooming and panning navigation, Pad and Pad++ also introduced three key concepts of interacting with ZUIs: semantic zooming, portals, and lenses (*Bederson, Hollan, Perlin et al. 1996*). Later sections of this chapter illustrate these concepts in greater detail.

Bederson et al. also mention that their ZUI could be used for mobile devices such as PDAs or in home entertainment devices such as interactive set-top cable boxes. It took almost 10 years until this became true for mobile devices, e.g., in (*Bederson, Clamage, Czerwinski et al. 2004*; *Büring, Gerken, and Reiterer 2006*), and except our ZOIL-based *EuroITV* demo (*Jetter, Engl, Schubert et al. 2008*), I am only aware of a single commercial product from

Hillcrest Labs[45] that employs ZUIs in the domain of interactive television today. Instead, ZUI designs and prototypes such as DENIM (*Lin, Newman, Hong et al. 2000*), PhotoMesa (*Bederson 2001*), DateLens (*Bederson, Clamage, Czerwinski et al. 2004*), HyperGrid (*Jetter, Gerken, König et al. 2005*), or Inspector (*Memmel 2009*) moved away from an ambitious top-down redesign of the GUI as ZUI. Instead, they used ZUIs as UI components inside the application windows of the dominant desktop metaphor. Furthermore, ZUI user studies rather than ZUI systems became the typical topic of ZUI research in HCI and InfoVis: Typical examples are the benefit of animations in ZUIs for building mental maps (*Bederson and Boltman 1999*), overview+detail vs. detail-only techniques (*Büring, Gerken, and Reiterer 2006*; *Hornbaek, Bederson, and Plaisant 2002*), or zooming vs. multiple windows (*Plumlee and Ware 2006*).

4.1.4 Commercial and Future ZUIs

In 2000, Jef Raskin, who was unaware of Pad and Pad++ (*Raskin 2000: 156*), (re)suggested replacing the *"mazelike"* desktop metaphor and its application windows with an infinite plane of information having infinite resolution: *"The zooming interface paradigm can replace the browser, the desktop metaphor, and the traditional operating system. Applications per se disappear"* (*Raskin 2000: 164*). In light of the great advances in graphics and network performance, his vision appears feasible today:

First, popular applications such as Google Maps or Google Earth and novel technologies such as Microsoft's zoom.it or Deep Zoom[46] enable ZUI navigation in vast server-side tera- or peta-pixel images – even inside a mobile Web browser. This technology is not only used for navigating imagery or maps but also abstract information spaces with Microsoft's PivotViewer control[47].

Second, UI toolkits and frameworks from HCI research such as Jazz and Piccolo (*Bederson, Grosjean, and Meyer 2004*) successfully used structured scene graphs to enable a more efficient ZUI programming. Since then, scene graphs have been adopted by commercial UI platforms such as Windows Presentation Foundation (WPF) that provides hardware-accelerated rendering of vector-based UI elements and matrix transformations for scaling UIs without pixelation. Based on such new commercial UI platforms, researchers and software developers have begun to create many novel ZUI-based tools, e.g., Code Canvas, a prototype of a zoomable frontend for Microsoft Visual Studio (*DeLine and Rowan 2010*).

Third, after moving from the room-sized special purpose hardware of SDMS to standard desktop PCs, ZUIs are now moving beyond the desktop into the post-WIMP age. Zooming and panning is extensively used on mobile devices, large screens, or tabletops where using traditional scrollbars and precise pointing would be slow or impossible (*Raskin 2000: 154*). Today, ZUIs have become a part of the design guidelines for future natural

[45] Hillcrest Labs: *http://hillcrestlabs.com* (Accessed Jan 4th, 2012)
[46] Deep Zoom: *http://msdn.microsoft.com/en-us/library/cc645050(v=VS.95).aspx* (Accessed Jan 4th, 2012)
[47] PivotViewer: *http://www.microsoft.com/silverlight/pivotviewer/* (Accessed Jan 4th, 2012)

user interfaces (NUIs) for touch and gesture. They are used to replace traditional hierarchical navigation and to leverage users' spatial memory (*Wigdor and Wixon 2011*). ZUIs on touch displays, tabletops, and tablets are also used in commercial zoomable presentation solutions for exhibitions, meeting rooms or TV studios (Figure 69).

Figure 69 – ICT smartPerform is a zoomable presentation solution for post-WIMP and multi-touch devices by ICT AG, Kohlberg, Germany. It was created by my former colleagues Werner A. König and Jens Gerken and was inspired by our joint work on ZUIs and ZOIL. Source: http://www.smartperform.de.

4.2 ZUI Foundations

To enable an unambiguous discussion of ZUIs and ZUI concepts, this section introduces some operational definitions and terminology that also enable a mathematical analysis of ZUI navigation. This is necessary, since the seemingly simple nature of a ZUI from a user's perspective hides some non-trivial complexity that plays a decisive role when implementing and studying ZUIs.

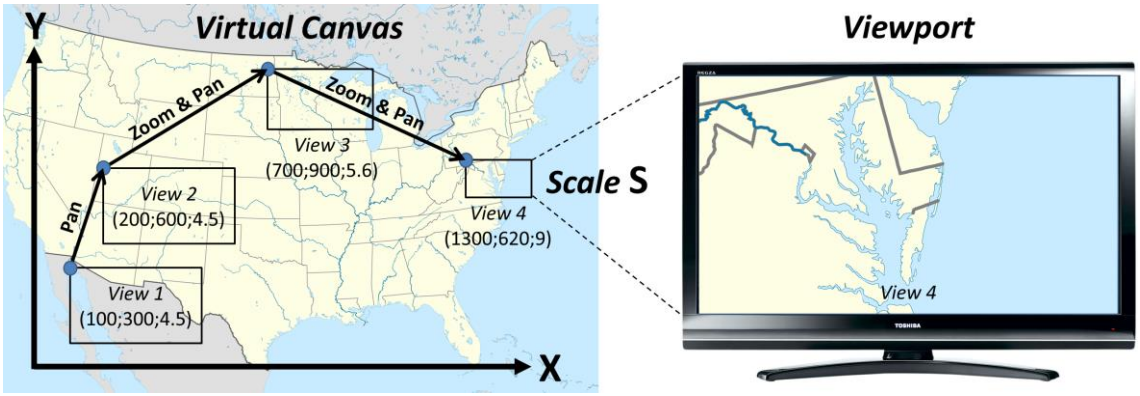


Figure 70 – ZUI Foundations: *Canvas*, *viewport* and examples of different *views* with their *X,Y,S*-coordinates.

A ZUI or 'zooming & panning' UI relates two parts: a *viewport* and a large virtual *canvas* (Figure 70). The *viewport* is the region of the screen that hosts the ZUI. It receives events

from the input devices, e.g., mouse and touch, and continually renders a part of the *canvas*. Thus it serves as a window through which the user can view and interactively explore the spatially distributed data that is contained in the *canvas*. The part of the *canvas* that is currently visible inside the *viewport* is called the *view*. The current *view* can be specified using three coordinates: The top-left corner of the *view* in *canvas*-coordinates X and Y and the magnification or scale-factor S. S determines the size of the *view* in relation to the size of the *canvas*: The greater S, the smaller is the part of the canvas that the *view* covers, but the greater is the rendered level of detail. $S = 1.0$ typically means that the *view* covers the entire *canvas*. Figure 70 illustrates different *views* and their coordinates.

To explore the *canvas* of a ZUI, users can employ two interaction primitives: *panning* and *zooming*. *Panning* means changing X and Y without changing S. To the user, *panning* feels like moving sideways in the *canvas*. *Zooming* means changing S without changing X and Y. To the user, *zooming* feels like approaching or stepping back from the *canvas*. A 'panning UI' or 'scrolling UI' can be defined as a special case of a ZUI where only panning is allowed and $S = const$ with $S > 1.0$.

For implementing ZUIs, it is often necessary to calculate an animation path between a starting point $A(x_a, y_a, s_a)$ and a destination point $B(x_b, y_b, s_b)$. For example, clicking on a very small icon in an overview like Figure 63 (left) should result in an automated zoom operation from the overview into the content of a document (Figure 63, bottom right). Thereby such an automated zoom operation pans from the center of the canvas to its very edges and increases the scale factor by 100 or more. In comparison to jumping the user immediately to the new view, animating panning and zooming can dramatically reduce the users' cognitive load (*Cockburn, Karlson, and Bederson 2008*). However, as Cockburn et al. point out, implementing such zoom animations is also easy to do badly and fine-tuning the duration of the animation is important. Ideally, an animation's path is direct, monotonic, and short, so that the animation does not use an unnecessarily complex motion in space or takes too long in time. On the other hand, the animation should also be extensive enough in space and time to be readily perceived and spatially comprehended by users, so that they can build better mental maps of the canvas (*Bederson and Boltman 1999*). Therefore, calculating a "good" animation path and choosing an appropriate animation speed is important and not trivial.

The reason for the complexity of calculating good animation paths in a ZUI is their non-Euclidean nature. Unlike in a 3D Euclidean space with X-, Y-, and Z-coordinates, movements in a ZUI have a linear effect in the XY-plane while they are logarithmic in S. Furnas and Bederson discuss this as the 'joint pan-zoom problem' (*Furnas and Bederson 1995*). In a ZUI, the length of a path is highly dependent on the movement in S and not only in X and Y: "*A vast distance may be traversed by first zooming out to a scale where the old position and new target destination are close together, then making a small pan from one to the other, and finally zooming back in (...). Since zoom is naturally logarithmic, the vast separation can be shrunk much faster than it can be directly traversed, with*

exponential savings in the limit" (*Furnas and Bederson 1995*). This is also discussed by (*Van Wijk and Nuij 2003*). They consider an animation between two views in a ZUI as optimal when it is *smooth and efficient* and they provide an analytic solution for such optimal animations and for calculating their path length. However, their approach involves complex mathematics including differential geometry, solving a system of three differential equations and two free parameters that need user experiments to find good values. In this thesis and in ZOIL, I therefore use a simpler approach that is based on Furnas and Bederson's 'space-scale diagrams' to calculate direct and monotonic animation paths (*Furnas and Bederson 1995*). The following sections discuss this approach.

4.2.1 Animated Pan-Zoom Trajectories

Initially, creating a ZUI animation (or 'pan-zoom trajectory') between a starting point A and a destination point B might appear trivial. However, as Furnas and Bederson point out, it is not possible to simply compute the linear pans and log-linear zooms separately and execute them in parallel: *"The problem is that when zooming in, the world view expands exponentially fast, and the target point (...) runs away faster than the pan can keep up with it. The net result is that the target is approached non-monotonically: it first moves away as the zoom dominates, and only later comes back to the center of the view. Various seat-of-the pants guesses (taking logs of things here and there) do not work either"* (*Furnas and Bederson 1995*). Furnas and Bederson's space-scale diagrams reveal a hyperbolic relationship between scale factor and panning for monotonic pan-zoom trajectories. In the ZOIL software framework, the calculation of a pan-zoom trajectory between a point $A(x_a, y_a, s_a)$ and $B(x_b, y_b, s_b)$ follows this hyperbolic relation. The current view coordinates X and Y are animated along a path between start coordinates (x_a, y_a, s_a) and destination coordinates (x_b, y_b, s_b). For each intermediate point (X, Y) the scale factor S is calculated as a function of (X, Y). Furnas and Bederson provide a simplified one-dimensional formulation. For the ZOIL software framework, I enhanced their formula to implement a generic two-dimensional version that also handles the special cases of animations parallel to the X or Y axes where the calculation of m_x or m_y becomes impossible.

$$S = \begin{cases} \dfrac{s_a - m_x\, s_a\, x_a}{1 - m_x\, X} \text{ where } m_x = \dfrac{s_b - s_a}{s_b\, x_b - s_a\, x_a} \text{ when } x_a \neq x_b \text{ and } y_a = y_b \\ \dfrac{s_a - m_y\, s_a\, y_a}{1 - m_y\, Y} \text{ where } m_y = \dfrac{s_b - s_a}{s_b\, y_b - s_a\, y_a} \text{ when } x_a = x_b \text{ and } y_a \neq y_b \end{cases}$$

If $x_a \neq x_b$ *and* $y_a \neq y_b$, the calculation of S can be taken out either with m_x or m_y and there is no need for a distinction. If $x_a = x_b$ *and* $y_a = y_b$ (i.e., pure zooming along the S axis without a panning motion) the above formula is unnecessary and it is sufficient to only animate S from s_a to s_b.

4.2.2 Non-linear Space-Time Functions

To achieve a more natural and attractive motion during the animation from (x_a, y_a, s_a) to (x_b, y_b, s_b) over time, ZUIs should make use of non-linear space-time functions. During my

informal experiments, a simple linear space-time function without acceleration or deceleration appeared unnatural, unattractive and also harder to follow and comprehend for users. Instead, non-linear space-time functions can for example decrease the time span that is spent close to (x_a, y_a, s_a) and increase the time that is spent close to the destination (x_b, y_b, s_b). For the users, this creates a sensation of an immediate reaction since the animation starts immediately and moves fast right after its start, but as soon as the destination is approached, the movement gradually slows down which results in a more natural and physical motion. The ZOIL framework uses such an elliptic space-time function as default. Other variants such as a cosine or polynomial function are available or can be easily implemented (Figure 71).

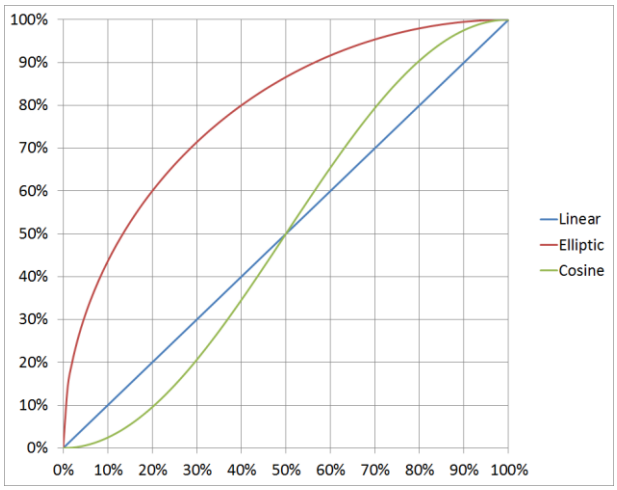


Figure 71 – Three different space-time functions in ZOIL. On the X-axis: percentage of time of the animation. On the Y-axis: percentage of total distance to pan between start and destination. While linear and cosine movement appeared unattractive, elliptic movement was most appealing and is ZOIL's default.

4.2.3 ZUI Animation Duration

Apart from its trajectory and non-linear space-time function, the total duration of a ZUI animation and its zooming speed have an effect on attractiveness, comprehensibility, and user performance. Past user studies of zooming speeds in ZUIs suggest that user performance is best with a zooming speed of 8 scale factors per second (*Guo, Zhang, and Wu 2000*). Also, 4 and 12 factors per second still showed reasonably good results while slower or higher speeds decreased task performance and subjective satisfaction. However, other research suggests that constant animation times between 0.3 and 1.0 seconds are appropriate (*Cockburn, Karlson, and Bederson 2008*). Based on these inconclusive results, I informally tested different functions and parameters for calculating the animation duration in the ZOIL framework. As a result, the ZOIL default implementation determines the duration T for a zoom-pan animation between two views at different scale factors s_a and s_b as follows:

$$T = \frac{D}{d} + T_{min} \text{ where } D = \begin{cases} \frac{s_a}{s_b} \text{ for zooming out} \\ \frac{s_b}{s_a} \text{ for zooming in} \end{cases}$$

Thereby d is the default zooming speed in scale factors per second (default value $d = 35$) and T_{min} is the minimum duration of a zoom animation (default value $T_{min} = 1\,\text{s}$). Interestingly, for the ZOIL framework, the default value for d turned out to be three to four times higher during informal experiments than the speed recommended by (*Guo, Zhang, and Wu 2000*). Zooming speeds below 35 scale factors per second felt unnecessarily slow. Nevertheless, Guo et al.'s recommendation of a fixed zooming speed instead of a fixed animation duration as in (*Cockburn, Karlson, and Bederson 2008*) proved to be helpful, in particular for cases of very small or great values of D where a fixed duration either resulted in unnecessarily slow animations or in far too rapid zooms that were too hard to perceive and comprehend.

4.2.4 Length of a ZUI Animation Path

A further challenge of ZUIs is measuring the distance that a user has travelled in a ZUI. This is particularly important for user studies and for evaluating the efficiency of ZUI navigation. To calculate the length of an animated or manually executed pan-zoom trajectory, I use an information-based metric that calculates the 'cost' for navigation using a further suggestion of Furnas and Bederson. They suggest that the cost of navigation should depend on the amount of visual information that has to be added into a viewport during zooming and panning and thus can also be seen as a measure for the additional visual information that has to be processed by the user. *"The suitability of the information-based approach followed here hinges on an implicit cognitive theory that humans watching a pan/zoom sequence have somehow to take in, i.e., encode or understand, the sequence of views that is going by"* (*Furnas and Bederson 1995*). Furnas and Bederson used informal testing to see if there was an obvious preference between trajectories and compared these to their theory. Their overall conclusion was that the information metric, based on analysis of space-scale diagrams, is a reasonable way to determine good pan-zoom trajectories.

The metric is based on following assumptions: During panning, entirely new information moves into the viewport and that is 'expensive'. However, zooming always keeps parts of the current visual information and is therefore 'less expensive'. Due to this different nature of panning and zooming, the cost of a pure pan is linear in the distance panned, and the cost of a pure zoom is logarithmic with change of scale. Both costs are related using a constant that is determined by the number of pixels in the viewport (*Furnas and Bederson 1995*). While Furnas & Bederson formulate and explain this metric using an abstract one-dimensional example, its actual application for real-world ZUIs and user studies needs a two-dimensional formulation. Therefore, I formulated a two-dimensional variant of Furnas & Bederson's metric as follows:

A user executes a navigation step in a ZUI between time t_{n-1} and t_n. The distance travelled in the ZUI during this step is $\Delta x, \Delta y, \Delta s$ where $\Delta x = |x_n - x_{n-1}|$, $\Delta y = |y_n - y_{n-1}|$ and $\Delta s = \frac{s_n}{s_{n-1}}$. Here, $x_n, x_{n-1}, y_n, y_{n-1}$ are screen coordinates that can be calculated from

the canvas coordinates X, Y. Given a ZUI with a viewport size of $W \times H$ (e.g. 1024 x 768 pixels), the cost c of the navigation step can then be formulated as:

$$c = H\Delta x + W\Delta y - \Delta x\Delta y + WH|\log \Delta s|$$

Figure 72 illustrates the costs of panning and zooming. $H\Delta x + W\Delta y - \Delta x\Delta y$ is the total amount of pixels that have to be updated due to the panning motion. $H\Delta x$ and $W\Delta y$ are the amounts of pixels that are panned into the viewport. Since horizontal and vertical panning happens in parallel and not separately, there is an overlap of $\Delta x\Delta y$ pixels that is only loaded once. Therefore, the cost for one update is subtracted.

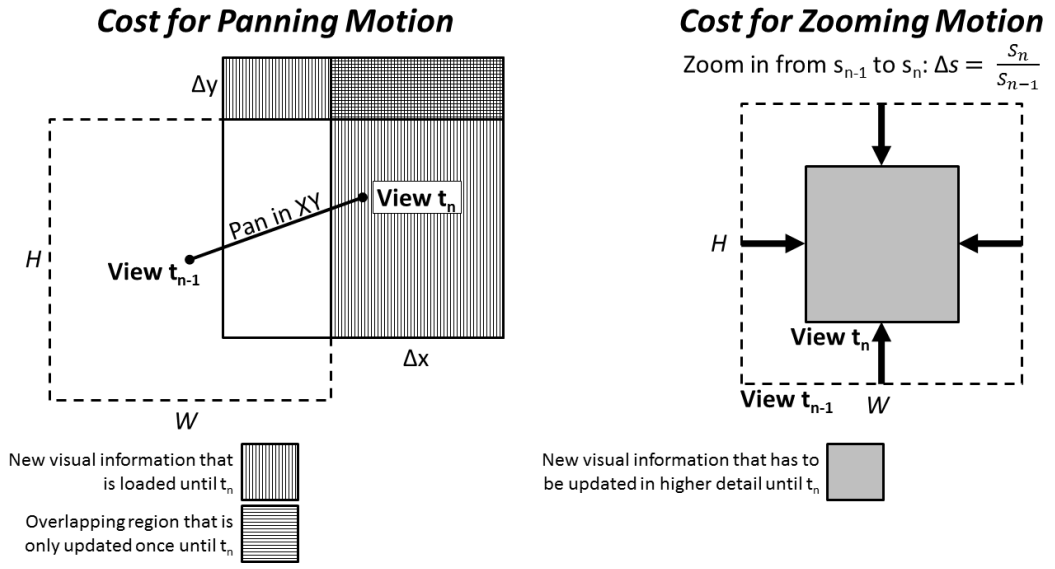


Figure 72 – Illustration of the cost metric that is used for measuring the length of zoom-pan trajectories in ZOIL and this thesis.

The cost of zooming can be approximated by considering a zoom operation as an infinite number of incremental zoom steps where each step leads to loading new visual information. In the case of zooming in, the original lower-resolution visual information has to be refined by adding new pixels with higher-resolution information in between (Figure 72 right). In the case of zooming out, the original higher-resolution information is shrunk below the original window size and the resulting empty space surrounding the shrunk region has to be filled with new lower-resolution visual information. Following the model of (*Furnas and Bederson 1995*), $\log \Delta s$ is the continuous limit of the cost of all these hypothetical intermediate zoom steps $\Delta s_1 + \Delta s_2 + \cdots + \Delta s_n = \Delta s$ where $n \to \infty$. As a result, $WH|\log \Delta s|$ is the total amount of pixels to be updated during a magnification change of Δs.

The unit of the resulting ZUI navigation cost metric is 'square pixels' which may appear surprising at first, but is simply because 'pixels' are used as a unit of length here and the resulting cost is measured in total updated screen area. In my research, this cost metric proved to be especially helpful in the user study on the effect of mouse vs. multi-touch on spatial memory (see section 4.5.5).

4.3 Interacting with ZUIs

Navigation in a ZUI is a highly interactive process during which users have to control the current view position to arrive at their intended destination in the canvas. This involves close cycles of perceiving and relating the current view and its position with the destination location as it is stored in the users' spatial "mental model" or "cognitive map" of the canvas. To achieve a direct manipulation of the view with a narrow gulf of evaluation and execution in the sense of (*Hutchins, Hollan, and Norman 1985*), the visual output including global and relative landmarks must serve the users' orientation by providing cues for the current scale factor or indicating neighboring objects or regions. Thus a good design of the visual output reduces what Hutchins et al. refer to as 'semantic distance'. Similarly, the input device and the mapping of its input signals to panning and zooming operations is particularly important for reducing Hutchins et al.'s 'articulatory' distance. For example, designers have to decide whether pushing the mouse button and moving the mouse to the left results in a pan to the right (as if the mouse "grabs" and moves the canvas) or a pan to the left (as if the mouse moves the virtual camera of the view).

4.3.1 Mouse and Multi-Touch Manipulations for ZUI Navigation

Over the years, HCI researchers have used and evaluated many different input devices for ZUIs (e.g. bi-manual controllers, pressure-sensitive stylus) and they experimented with different mappings of user input to panning and zooming operations (*Cockburn, Karlson, and Bederson 2008*). Today, the two most popular and relevant input devices for ZUI navigation are mouse and multi-touch and the different mappings have converged to dominant designs that I describe in the following section.

Panning with Mouse or (Multi-)Touch

Panning with the mouse typically uses the metaphor of "dragging" the canvas, e.g., in Google Maps, Bing maps, or Microsoft Deep Zoom. The mouse cursor can be moved freely inside the viewport. By pressing and holding the mouse button over an arbitrary location in the viewport, the underlying canvas is "grabbed" at this location and can be "dragged" around by moving the mouse. Moving the mouse to the left means dragging the canvas to the left and results in a pan of the view to the right. Thereby the canvas is typically "sticking" to the mouse cursor during dragging. This means that the view is always panned exactly by the distance by which the mouse cursor is moved in the viewport. Panning with touch or multi-touch is very similar to panning with the mouse, but instead of pressing the mouse button and moving the mouse, users touch the canvas in the viewport with one or multiple fingers and slide them (Figure 73).

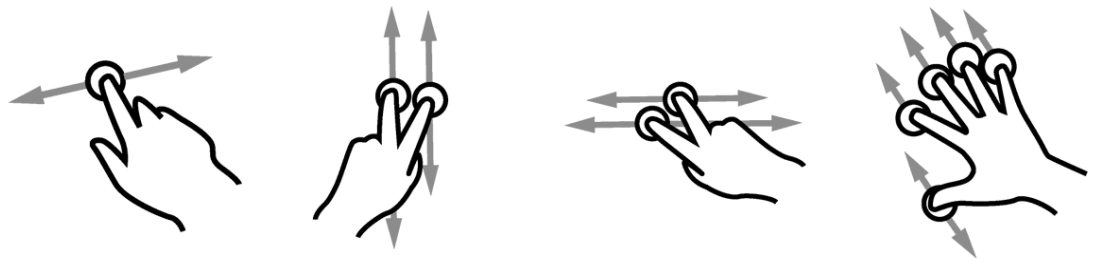


Figure 73 – Different variants of panning with (multi-)touch.
Adapted from *http://gestureworks.com*.

Zooming with Mouse or (Multi-)Touch

The simplest and most efficient way to navigate in a ZUI with mouse and touch is to single click or tap the destination object (or 'zoom target') in the viewport to issue an automated zooming and panning animation. As described above, this animation monotonically zooms and pans until the zoom target appears enlarged and is centered in the viewport. However, there are also situations in which a manual control of the scale factor S is desirable. For example, this can be used for smooth adjustments of the view or for zooming to destinations that do not contain designated zoom targets.

When using a mouse, manual zooming is typically done with the mouse wheel. The direction and amount by which S is changed is controlled with a stepwise or continuous rotation of the mouse wheel. Its extent is counted in ticks or degrees of rotation. Rotating the mouse wheel by a tick or degree in a forward direction leads to an increase of the scale factor S and thus a zoom-in. Rotating the mouse wheel by a tick or degree in a backwards direction leads to a decrease of the scale factor S and thus a zoom-out. In the ZOIL software framework, the new scale factor S_n for each tick is calculated by multiplying the old scale factor S_{n-1} with a constant w. The default value for w is 0.95.

$$S_n = \begin{cases} \frac{1}{w} S_{n-1} & \text{for a forward tick (zoom in)} \\ w\, S_{n-1} & \text{for a backward tick (zoom out)} \end{cases}$$

While early ZUI designs ignored the current mouse cursor position during zooming in or out, more recent designs use the mouse cursor to specify the *"center of zooming"* (*Raskin 2000: 154*) or *"zooming reference point"* (ZRP) (*W. A. König 2006: 36*). The ZRP is the center of a zooming operation, e.g., the point in the canvas into which is zoomed or from which is zoomed out. Thus, unlike older ZUI designs that always used the viewport's center as a fixed ZRP, newer designs ensure a dynamic control of the ZRP with the mouse cursor. This way, users can precisely control the center of the zooming operation to zoom into objects that are not in the center of the viewport but closer to its edges without panning. During zoom-out, this helps to reduce the amount of panning needed to keep the zoom-out's starting point visible inside of the viewport. Changing the ZRP by moving the mouse cursor during an ongoing zoom operation also enables users to continuously adjust or even change their zoom destination. This gives the user a sensation of a more

direct control of the zooming and facilitates smooth and efficient navigation without interruptions or mode switching.

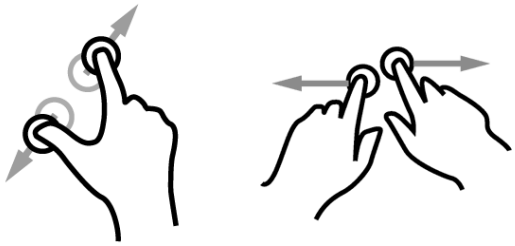


Figure 74 – Zooming using the 'pinching gesture' (left) or 'two finger zoom' (right). Adapted from *http://gestureworks.com*.

Zooming with multi-touch is typically done using the so-called 'pinching gesture' or 'two finger zoom'. Although 'pinching' is often attributed to the Apple iPhone, early uses of similar interaction can already be found in the seminal tabletop system metaDESK from the MIT Media Lab (*Ullmer and Ishii 1997*). Multi-touch enables users to control the scale factor *S* either with a one-handed thumb-and-index-finger or a bi-manual two-finger gesture (Figure 74). By touching two points in the canvas and simultaneously dragging them apart, the user cannot only specify the direction of changing *S* (e.g., zooming in), but also the precise absolute amount by which *S* is changed (e.g., zooming in by factor 4.23). For example, if users intend to zoom into a small object, they can touch the object at opposite corners and slide their fingers apart to zoom in until the object is displayed exactly in the desired size. Thereby the originally touched corners always remain directly under the fingers. This absolute control of *S* is different from the relative control of *S* with a mouse wheel where the direction and amount by which *S* is changed can only be specified tick-wise.

Apart from a precise absolute control of *S*, the pinching gesture or two-finger-zoom further enables users to freely choose and adjust the ZRP which always lies in the middle between the touch points of the fingers. Furthermore, sliding of the fingers without changing their relative distance is typically mapped to panning. As a consequence, users can use multi-touch manipulations for a combined zooming & panning that simultaneously zooms the destination object to any desired size and also pans it to any desired screen location.

Compared to traditional designs, an advantage of all the above-mentioned mouse and multi-touch manipulations is the user's ability to control zooming and panning simultaneously without interruptions for switching modes. Instead of having two modes for zooming and panning, users can combine both operations in a single zoom-pan trajectory. As a consequence, the usability and efficiency of the ZUI is increased by enabling users to manually approximate an "optimal" zoom-pan trajectory and ideally achieving the "exponential savings" of ZUI navigation that Furnas and Bederson discuss (*Furnas and Bederson 1995*).

Rotation with Multi-Touch

For some multi-touch ZUIs, it is also desirable to enable users to rotate objects or the entire canvas, e.g., for around-the-table collaboration when using maps or similar on a tabletop. To this day, the ZOIL software framework does not provide a rotatable information landscape and such a feature is also not implemented in any of the ZOIL-based prototypes. However, if desired, all individual objects in ZOIL's information landscape can become rotatable and users can rotate them by grabbing them in their corners or in their center (Figure 75). Similar to metaDESK (*Ullmer and Ishii 1997*), ZOIL-based user interfaces can thus combine zooming, panning, and rotation in a single multi-touch manipulation.

Figure 75 – Different variants of rotation with multi-touch.
Adapted from *http://gestureworks.com*.

4.3.2 Other Devices for ZUIs

In some cases, the previously described mouse or multi-touch manipulations cannot or should not be used as interaction techniques. One of these cases is our ZOIL-based demonstrator *EuroITV* that illustrated how ZUIs can be employed in the context of interactive television (*Jetter, Engl, Schubert et al. 2008*)[48]. The demonstrator featured a ZUI for basic personal information management of e-mails, photos, notes, and video-on-demand movies. The intention during its design was to showcase the potential of ZOIL and ZUIs for high-definition digital TV sets or set-top boxes for interactive television. Since the physical setting was a living room with a distant large screen (Figure 76), it does not rely on touch or mouse input, but provides the Nintendo Wiimote controller as an input device similar to standard TV remote controls.

EuroITV uses this controller as a remote pointing device and maps the different buttons on the necessary input operations. Figure 77 illustrates the functionality and input mapping of the demonstrator. The cursor position can be controlled by pointing the Wiimote at the screen and moving it using hand and arm movements. Pushing the "A" button over a zoom target results in an animated zoom-in. Pushing the "A" button twice results in an animated zoom-out to the enclosing zoom target. To drag objects, users

[48] The *EuroITV* demonstrator and its interaction design were conceived of by me to illustrate the functionality of the ZOIL software framework. The demonstrator was implemented by Andreas Engl and previously appeared as a part of his Bachelor's thesis (*Engl 2008*). I authored a demo paper about *EuroITV* that was published at EuroITV (*Jetter, Engl, Schubert et al. 2008*). Sören Schubert supported the visual design and the implementation of an input handler to connect Nintendo's Wiimote controller to the demonstrator.

push and hold “A” over the target to grab an object and drop it by releasing “A” again. During dragging, the accelerometer of the Wiimote tracks the rotation of the controller in the user’s hands around the pointing axis, so that objects could not only be dragged to different locations, but could also be tilted by different angles. The “D-Pad” 4-way directional controller is used for panning. For free zooming, the ZRP travels with the mouse cursor and users can use the “+” and “-“ buttons to zoom in and out.

Figure 76 - Physical setup of EuroITV demonstrator with Wiimote controller and distant 30" high-resolution display[49].

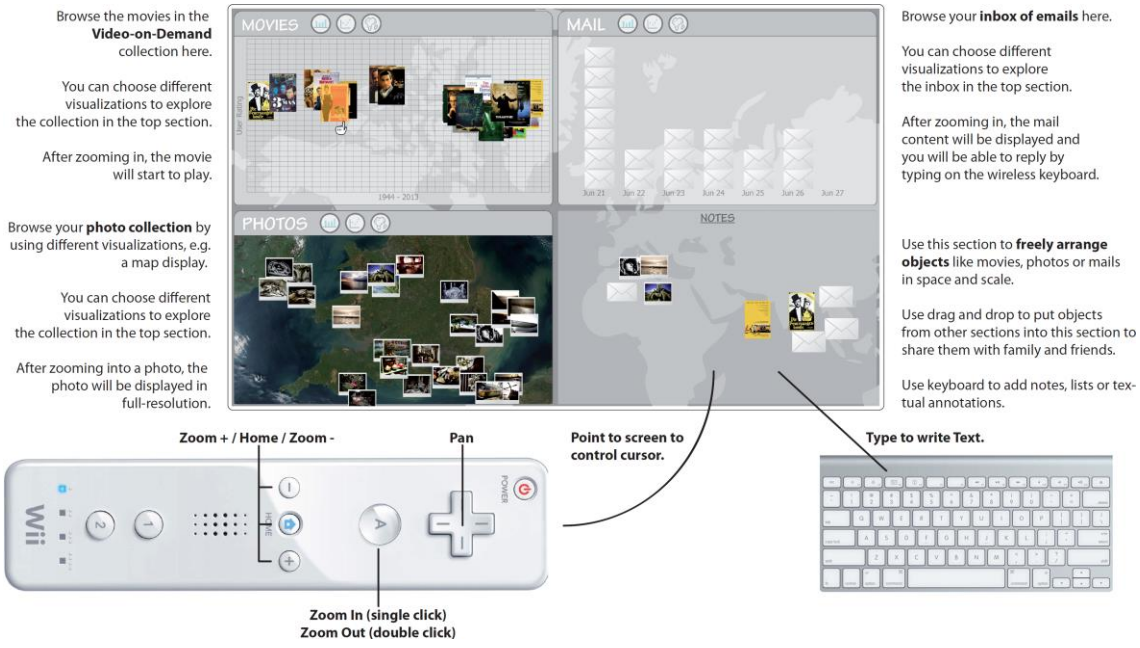


Figure 77 – Functionality and mapping of the *EuroITV* demonstrator.

In addition, the demonstrator also uses the “rumble” feature of the Nintendo Wiimote to provide the user with tactile feedback whenever the cursor on the screen enters a zoom target. For text input, *EuroITV* uses a small Apple wireless keyboard that easily fits on a user’s lap when sitting on a couch or around a coffee table in a living room.

[49] Taken from *EuroITV* video at *http://hci.uni-konstanz.de/jetter/euroitv.avi* (Accessed Sep 04, 2012)

4.3.3 Desert Fog

In some cases, free zooming and panning can result in a phenomenon called ‘desert fog’ that was first described by (*Jul and Furnas 1998*). Desert fog occurs when the ZUI user is faced with a situation where the immediate environment is totally devoid of navigational cues. For example, this happens when users zoom into a sparsely populated part of the canvas and arrive at a view position where there is not a single visible object in the viewport anymore. Since the viewport then only contains a magnified part of the empty canvas, users have no navigational cues that could aid their spatial orientation and help them to decide where to go from here. A similar desert fog situation can occur when users rapidly zoom out of a canvas, so that all objects shrink to a visual size that is close to or below the size of a pixel. In this situation, objects become invisible and are not rendered anymore.

To escape desert fog, the user might need to zoom out or zoom in until objects or other cues become visible again. An opposed strategy could be to stay at the current scale factor and to move sideways through the desert fog until new objects or cues are panned into the current view. However, it is hard to decide for the users which strategy to use, in particular if they cannot recall their whereabouts in the canvas or if they are not aware which previous inputs have brought them here. Furthermore, as long as there are no objects visible in the viewport, there is no visual feedback on the users’ input. This can become very confusing since users cannot recognize whether the system is still accepting their input and is executing any of their intended zooming and panning operations. Without visual feedback, the entire gulf of evaluation and eventually the users’ spatial orientation breaks down.

Using the ZOIL framework, it is possible to realize novel techniques that can help to avoid desert fog. One novel technique that I created is ‘parallax zooming’. The first design and implementation of this technique can be found in the above-mentioned *EuroITV* demonstrator (*Engl 2008: 28*). The technique uses a visual background that shines through the canvas in the foreground. This background always zooms and pans together with the current view on the canvas, albeit not at the same but at a smaller extent. When the foreground view is zoomed or panned, the background view does the same. However, the zoom factor and the panning distance applied on the background is only a fraction of that of the foreground.

The *EuroITV* demonstrator shows a world map that shines through the transparent canvas that contained the actual data (Figure 78). When the canvas in the foreground is zoomed or panned, the world map in the background does the same. However, the zoom factor and the panning distance applied on the background is only a fraction (e.g. $1/5^{th}$) of that of the foreground. This results in an always visible and slower moving world map as visual background that creates an illusion of depth similar to the ‘multiplane camera’ in traditional animation or ‘parallax scrolling’ in 2D video games. Therefore, I termed this technique *parallax zooming*. Due to its visibility and slower motion, the background always gives feedback whether zooming or panning is still executed even if the

foreground is empty because of desert fog. It also gives some indication on the absolute position of the view in relation to the entire canvas. An extension of this concept can now be found in recent versions of ICT smartPerform (Figure 69). There, smaller background objects pan at a slower speed than larger foreground objects, so that the objects themselves create an illusion of depth.

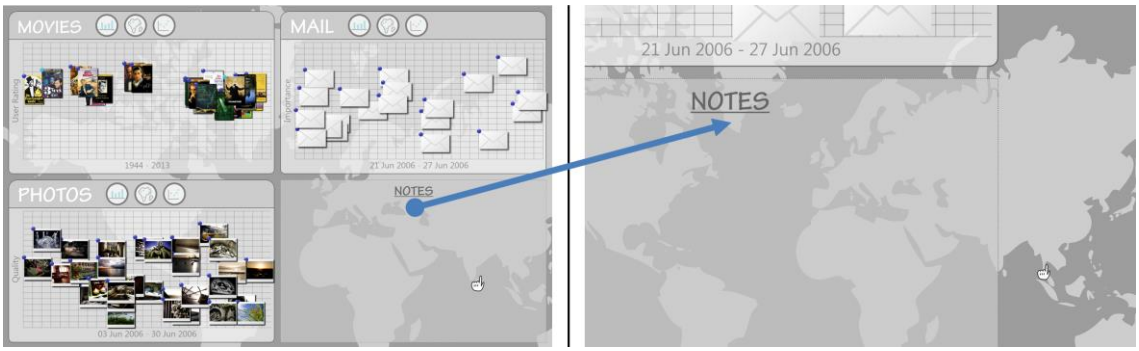


Figure 78 – *EuroITV* with parallax zooming. Left: The home screen with a canvas in the foreground and a world map in the background. Right: A zoom-in on 'Notes' also zooms and pans the map in the background. However, for the background, the scale factor and panning distance are only 1/5th of those of the foreground.

As described in section 4.5.5, desert fog can also be used to influence the users' navigation behavior. For example, in the experimental setup of our user study, we increased the distance between objects in the canvas to force users to use zooming more frequently and to prevent them from using 'brute-force' panning-only navigation strategies (*Jetter, Leifert, Gerken et al. 2012*).

4.3.4 Semantic Zooming

'Semantic zooming' (*Perlin and Fox 1993*) is an alternative to scaling ZUI objects purely geometrically. Instead, the object's visual representation is scaled in non-geometric ways that support user tasks better than plain geometric growth. During a semantic zoom, the growth of an object in display space is not just used to render the same object at a higher resolution, but also to reveal different and more content and functionality. For example, a document first appears as an icon and name, but during zooming in, it gradually grows and replaces its icon with thumbnails of the document's pages. When zooming in further, the object reaches a size where it covers enough display scape to replace the thumbnails with an editor application to view or edit the document on-the-spot without the need for any navigation in start menus or the file system. This smooth transition between iconic representation, metadata, and full-text/full-functionality reduces the need for managing overlaying application windows, menus, or pop-ups with details-on-demand. As mentioned in the previous chapter on OOUIs, this resonates with design principles like *"the content is the interface"* for natural user interfaces (*Hofmeester and Wixon 2010*) that recommend to focus UIs on their content and to use only a minimum of administrative controls or system states (*Wigdor and Wixon 2011: 31*). Particularly in the case of an OOUI, semantic zooming can be an efficient technique for providing the extended functionality that is attached to an object. Similar to the traditional "right-click

opens context menu”-design of the desktop metaphor, the functionality can be provided in direct spatial and logical proximity of an object but without using overlaying menus or dialog windows.

First concepts that are similar to semantic zooming can be traced back to the MIT’s SDMS system and to its “field version” that was installed on the US carrier Carl Vinson in 1980 (*Bolt 1984: 23*). This version enabled zoom navigation in a database of ships. When zooming into a ship’s symbol, more textual information about its fuel, speed and commanding officers appeared inside of it. The first official use of the term ‘semantic zooming’ can be found in the Pad system from 1993 that also introduced seamless transitions between different representations using fading (Figure 79).

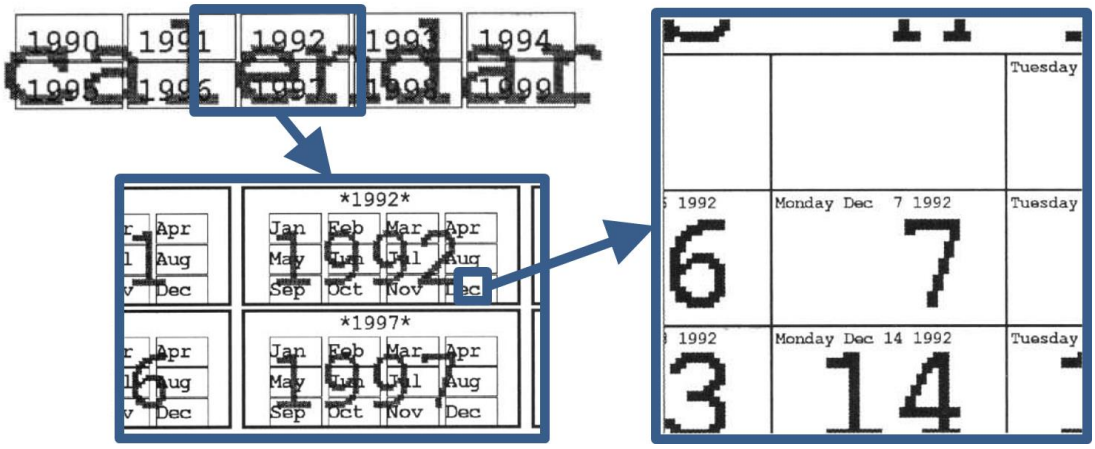


Figure 79 – Semantic Zoom into a calendar with Pad. When zooming into the year or month, the large scale display items gradually fade out and disappear and new smaller scale display items appear. Adapted from (*Perlin and Fox 1993*).

4.3.5 Semantic Zooming in ZOIL

For any ZOIL-based user interface, semantic zooming is essential and the ZOIL software framework provides a mechanism for its implementation. ZOIL’s semantic zoom is based on selecting one of multiple pre-defined representations for an object depending on the currently available display space for rendering it. For each object, the ZOIL framework automatically selects the most fitting from the pre-defined representations and renders it to the screen. As soon as a transition between different representations takes place (e.g. during zoom-in our zoom-out), ZOIL executes this transition using an opacity animation that results in the illusion of visual continuity and viewing a single changing object. Harsh switching between representations is avoided. Figure 80, Figure 81, and Figure 82 show different examples of semantic zooming from ZOIL-based prototypes.

To design and implement a semantic zoom with ZOIL, designers do not require knowledge of C# but they can use an entirely declarative approach. Each ZOIL object for semantic zooming carries one or more different representations that are defined in WPF’s declarative XML-based XAML markup language. The necessary XAML code can be created and edited using visual XAML editors or design tools such as Microsoft Expression Blend. As discussed in the context of the MVVM pattern in 3.4.4, this allows a

separation of concerns between design in XAML and code in C# and thus enables an improved designer/developer workflow.

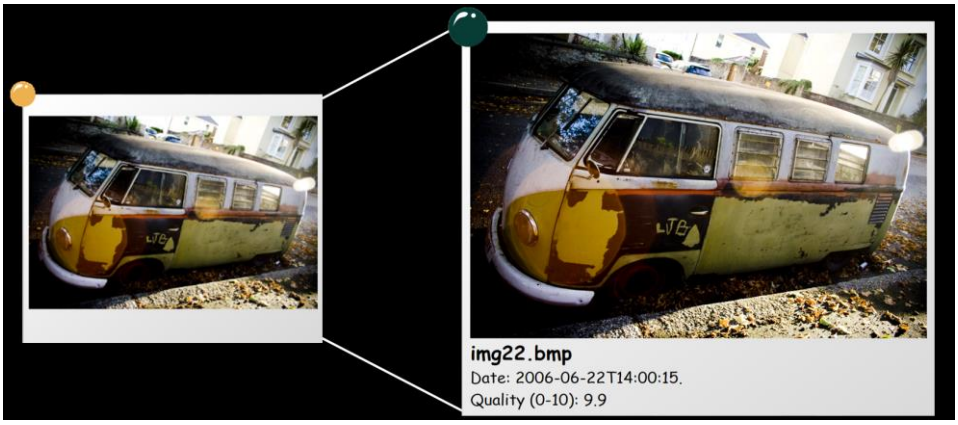


Figure 80 – A simple example of a semantic zoom from *EuroITV*. When zooming into a photo from a photo collection (left), a polaroid-like frame with additional metadata fades in (right).

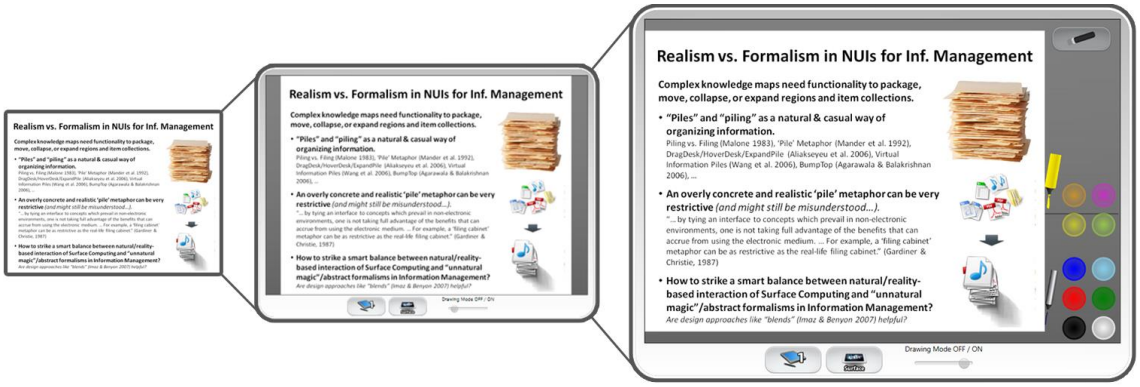


Figure 81 – Semantic zoom into a Powerpoint slide object in *DeskPiles*. During zooming, the object turns from a thumbnail of a slide into an editor with controls for sending the object to other devices and a tool palette for drawing and annotation.

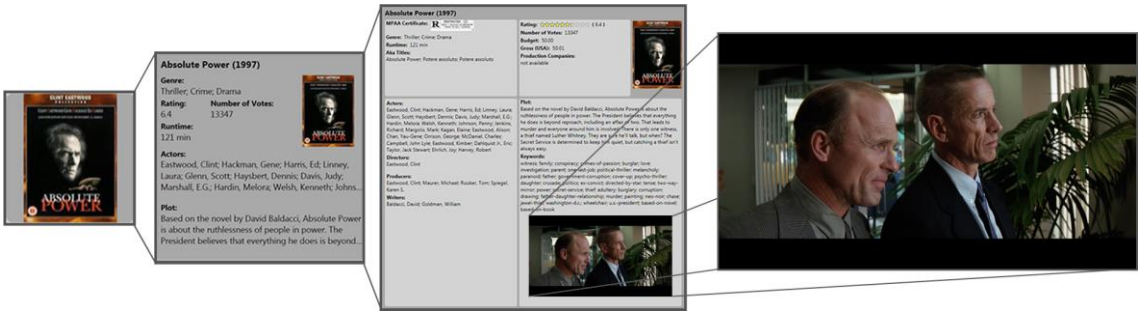


Figure 82 – Semantic zoom into a movie object from the *Media Seminar Room*: After the movie details fade in, the actual movie is displayed as a video stream. By further zooming into the stream, users can watch the movie in a full-screen view.

To provide a comprehensive support for authoring and implementing ZUIs, the ZOIL software framework extends the standard set of XAML elements and WPF's native controls with ZOIL-specific components. With this extended XAML code, designers and developers can define the object's visual appearance at different scales and also create

templates that define how values or content from databases or Web resources are fetched and displayed using data binding. Figure 83 illustrates this using a sample. It uses the three C# classes from the ZOIL software framework that are responsible for semantic zooming: *ZComponent* (derived from WPF's *ContentControl*), *ZComponentFrames* (derived from WPF's *Panel*) and *ZComponentFrame* (derived from WPF's *ViewBox*).

```
<ZComponent x:Class="MovieView">
 <ZComponentFrames>

  <!-- First representation -->
  <ZComponentFrame WidthNeeded="0">
   <Image Source={Binding PosterUri}/>
  </ZComponentFrame>

  <!-- Second representation -->
  <ZComponentFrame WidthNeeded="150">
   <DockPanel>
    <TextBlock Text={Binding Title}
               DockPanel.Dock="Top"/>
    <Image Source={Binding PosterUri}
               DockPanel.Dock="Left"/>
    <CustomView ItemsSource={Binding Actors}/>
   </DockPanel>
  </ZComponentFrame>

  <!-- Third representation -->
  <ZComponentFrame WidthNeeded="450">
   <MediaElement Source={Binding StreamUri}/>
  </ZComponentFrame>
```

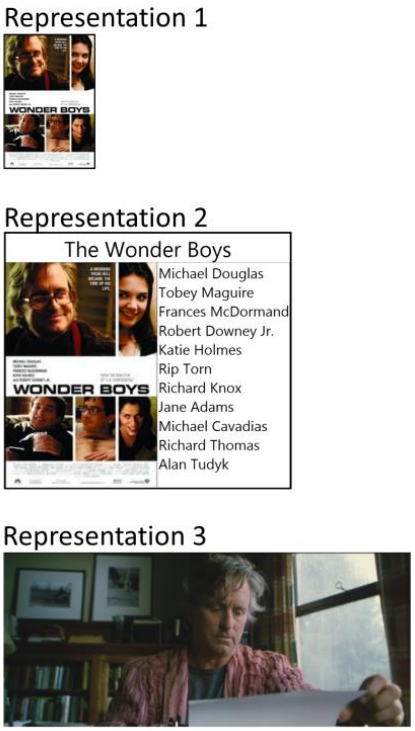


Figure 83 – Code sample in XAML. This code defines a basic movie object, similar to that in Figure 82. (For better readability some parts of the XAML code have been simplified).

Every class of UI objects in the zoomable information landscape that should support semantic zooming (e.g. *MovieView*) has to be derived from *ZComponent*. To achieve a semantic zoom, this class of UI objects also has to carry a *ZComponentFrames* object at the root of its visual appearance. This *ZComponentFrames* object contains the individual representations as *ZComponentFrame* objects and switches between them. It manages the visibility and the opacity of the different *ZComponentFrame* objects depending on the currently available render size and the different values of the *WidthNeeded* property that defines the size threshold for this representation.

In principle, the XAML definition of a *ZComponentFrame* is only static markup code. However, in almost every case, dynamic content from a database or the World Wide Web has to be a part of the visual representation of an object (e.g. the poster, actors names, or the video stream in Figure 83). In practice, the *ZComponentFrame* is therefore rather a template for content presentation than a static visual object. To enable designers

to easily integrate dynamic content into this template, the ZOIL framework makes use of the data binding feature of WPF. Inside the XAML definition, placeholders such as "{Binding Actors}" or "{Binding StreamUri}" can be used to bind list views or media players to string arrays, URIs, or database queries.

4.3.6 Portals and Multi-focus ZUIs

In Pad, Perlin & Fox also introduced 'portals' as a further concept for interacting with ZUIs: *"Portals are used for navigation, they are like magnifying glasses that can peer into and roam over different parts of the Pad surface. A portal may have a highly magnified view or a very broad, panoramic view, and this view can be easily changed. The screen itself is just a special 'root' portal"* (*Perlin and Fox 1993*).

Figure 84 illustrates how three portals can provide views and access to different scales and locations of the canvas in addition to the normal main view or 'root portal' that fills the entire screen. Thereby each portal serves as an independent viewport through which users can navigate the shared canvas and interact with the canvas' objects.

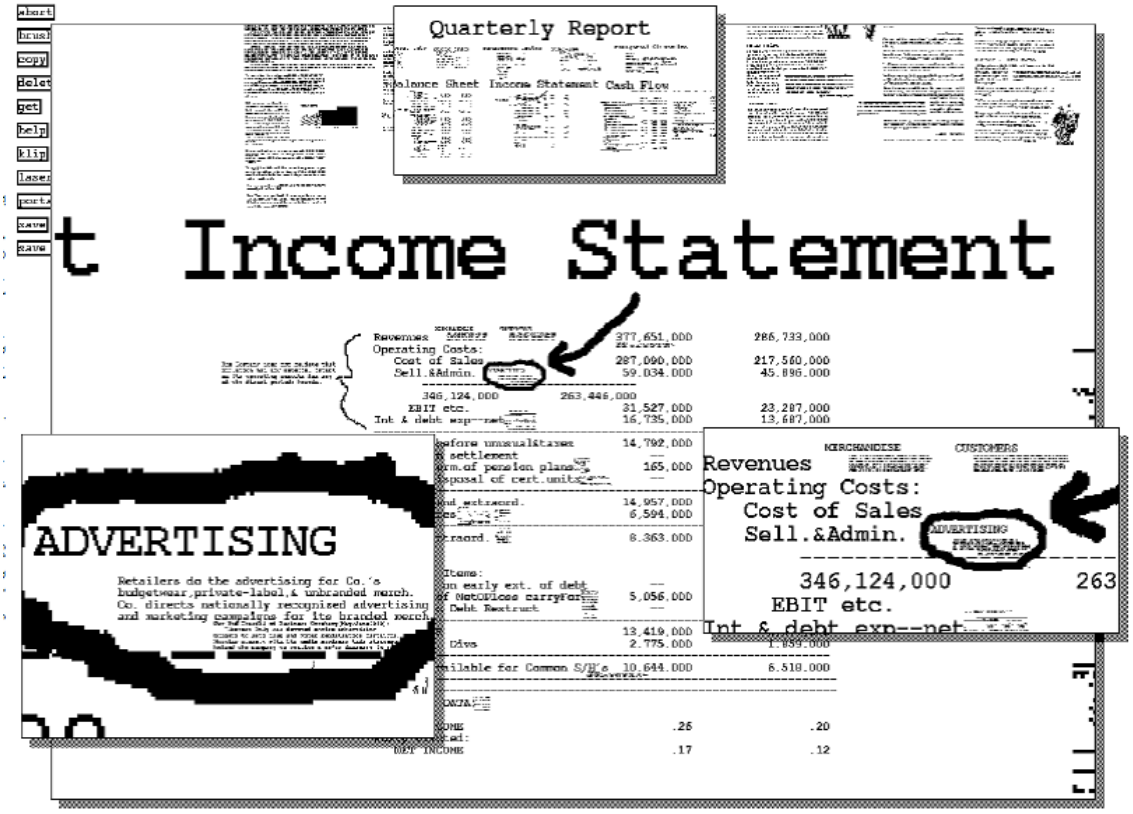


Figure 84 – Example of a quarterly report from Pad. The portals are views onto other parts of the report. Adapted from (*Perlin and Fox 1993*).

The purpose of portals is to let users create or destroy multiple simultaneous views, so that they can focus on different locations and scales of the canvas at the same time. Thus portals enable users to turn a single-focus ZUI into a multi-focus ZUI. As a result, users can for example create and configure different overview+detail views or they can put

distant regions of the canvas side-by-side on the screen, so that comparing objects or dragging objects between them becomes easier.

Subsequent work on Pad++[50], introduces more variants of portals and differentiates between *"non-sticky"* and *"sticky"* portals (*Bederson, Hollan, Perlin et al. 1996*). Non-sticky portals are attached to the canvas and pan and zoom with it. Sticky portals stick to the screen: *"Making an object sticky effectively lifts it off the Pad++ surface and sticks it to the monitor glass"* (*Bederson, Hollan, Perlin et al. 1996*).

In its current state, the ZOIL software framework does not natively support portals. However, Jonas Schweizer designed and implemented a multi-focus ZUI application with the ZOIL framework in his Bachelor's thesis that I supervised in 2009 (*Schweizer 2009*). The thesis also discusses how this implementation of a multi-focus approach on the application level can be integrated seamlessly into the existing ZOIL software framework. From a user's perspective, the suggested design is a simplified version of Bederson et al.'s sticky portals. The following figures demonstrate how the ZOIL-based multi-focus ZUI application was realized.

Figure 85 – A viewport can be split into two viewports (or sticky portals) to create two simultaneous views of different locations and scales. Each viewport enables users to freely navigate and interact. Objects can be moved between viewports using drag-and-drop. Source: (*Schweizer 2009*).

[50] See video of Pad++ by Ben Bederson at *http://www.youtube.com/watch?v=68lP1gRLmZw*

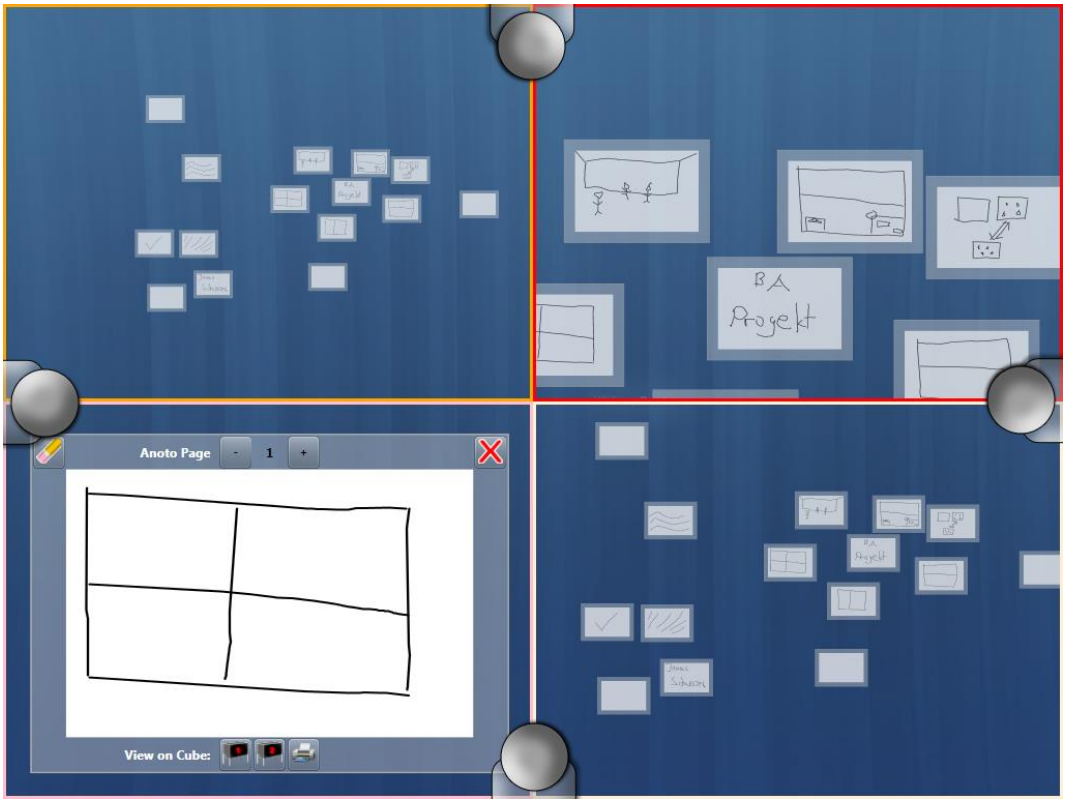


Figure 86 – The new viewport can be split several times. Source: (*Schweizer 2009*).

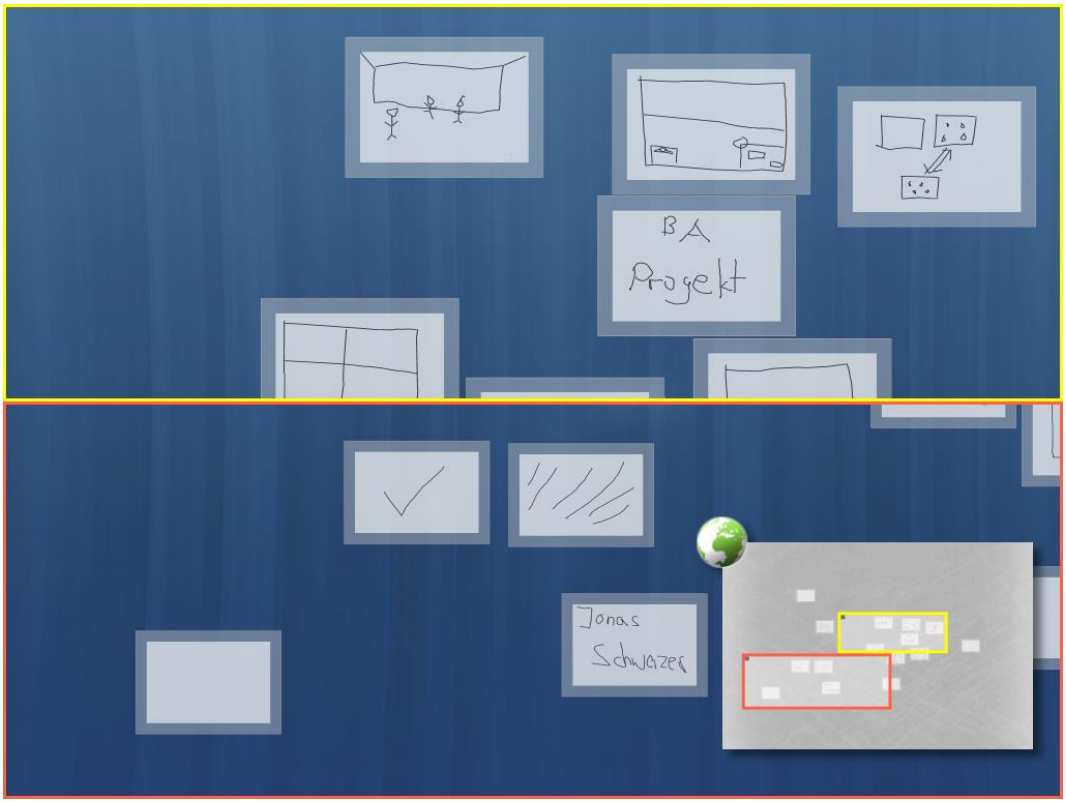


Figure 87 – Users can activate an additional sticky overview portal that floats above the viewports and shows which parts of the canvas are currently visible in which viewport. Source: (*Schweizer 2009*).

4.3.7 Lenses

Apart from portals, ‘lenses’ are another important concept of ZUI interaction. They were first introduced as ‘portal filters’ in Pad (*Perlin and Fox 1993*) and their concept was refined in Pad++ by Bederson et al.: *“Lenses are objects that alter appearance and behavior of components seen through them. They can be dragged around the Pad++ surface examining existing data. For example, data might normally be depicted by columns of numbers. However, looking at the same data through a lens could show that data as a scatter plot, or a bar chart”* (*Bederson, Hollan, Perlin et al. 1996*). Figure 88 shows examples of lenses in Pad++ that were inspired by the concept of the see-through ‘magic lenses’ of (*Bier, Stone, Pier et al. 1993*).

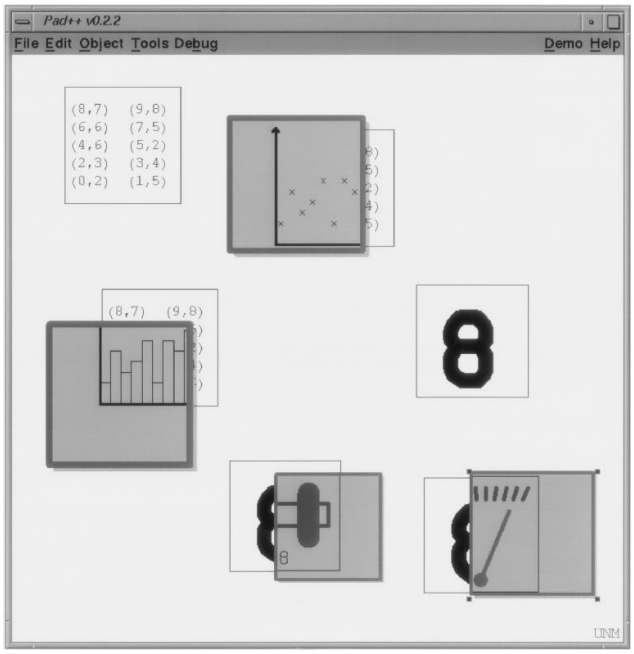


Figure 88 – Examples of lenses that show quantitative data as scatter plots, bar charts, sliders or scales. Source: (*Bederson, Hollan, Perlin et al. 1996*).

In a ZOIL-based UI, lenses are important tools for filtering and visualization purposes, especially when users have to manage many information objects with rich content. In these situations, searching for bits of information in the objects’ content or getting an overview of the characteristics of an entire cluster of objects becomes very challenging. Simply using repeated manual zooming and panning to browse manually between objects is too inefficient. Furthermore, today’s information objects typically carry extensive metadata like geo locations or user tags and ratings. For some tasks, a faceted navigation of such metadata using dynamic visualizations is far more efficient than manual browsing by zooming and panning into the users’ pre-defined visual hierarchies and locations.

For this reason, a ZOIL UI should provide lenses with information visualization tools for visually aggregating larger collections of information objects in alternative visual representations, e.g., maps, tables, plots, or cover flows. They can also provide controls for dynamic filtering of objects, e.g., range sliders or input fields for search queries. As is

discussed in greater detail in chapter 6, these visual tools enable more efficient search, filtering and analysis similar to visual information seeking systems such as *Film Finder* (*Ahlberg and Shneiderman 1994*).

The ZOIL-based prototype *MedioVis 2.0* by (*Heilig, Demarmels, Rexhausen et al. 2009*) makes extensive use of lenses (see section 3.5.4). Like in Pad++, rectangular lenses can be created by the user at any time and location and moved as a see-through overlay in front of the information landscape (Figure 89). These lenses serve as see-through filters or they visualize underlying item clusters using different visualization tools (Figure 90). By changing to the 'locked' mode, the user can also lock the current state and keep the content of a lens even when it is moved to other locations in the information landscape. Such 'locked' lenses can serve as containers for copying and persisting results from search or filtering (e.g. "new movies with rating > 6") and can be placed at arbitrary locations in the information landscape for later use.

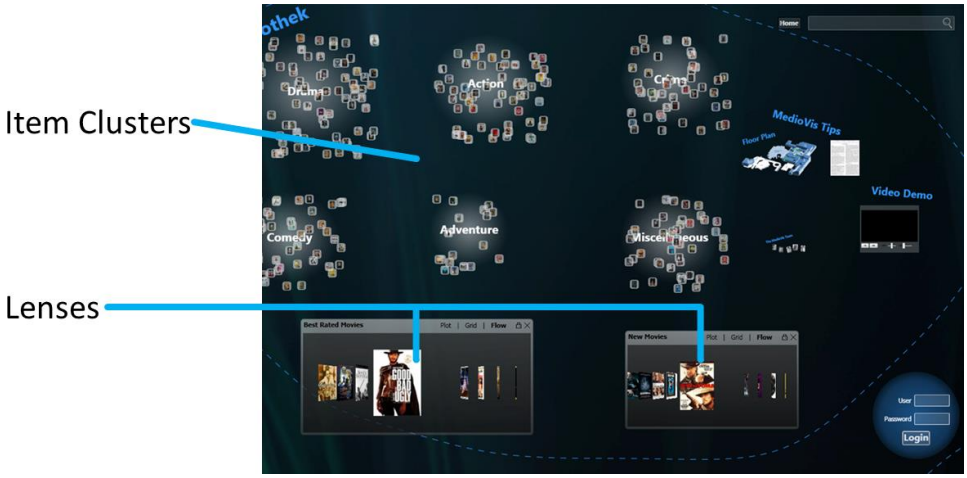


Figure 89 – Initial information landscape of *MedioVis 2.0* with different item clusters and two recommendation lenses. Adapted from (*Heilig, Demarmels, Rexhausen et al. 2009*).

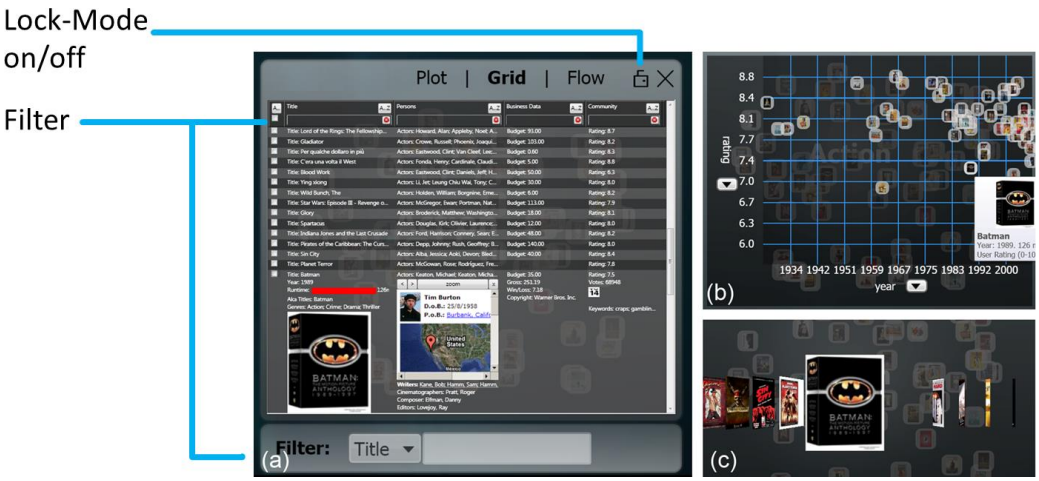


Figure 90 – Different visualizations and filters in MedioVis 2.0: (a) lens with HyperGrid (*Jetter, Gerken, König et al. 2005*), (b) lens with Scatter Plot, (c) lens with Cover Flow. Adapted from (*Heilig, Demarmels, Rexhausen et al. 2009*).

4.3.8 Tangible Lenses

The above-mentioned lenses in *MedioVis 2.0* are entirely virtual. They are created as pixels on the screen using pen input or a touch gesture. An alternative approach was explored in the *Media Seminar Room* and uses actual physical objects, e.g., small frames made from wood, cardboard or transparent foil that serve as “tangible lenses” (Figure 91). The first use of such tangible lenses for tabletops can be found in the metaDESK system from the MIT Media Lab (*Ullmer and Ishii 1997*).

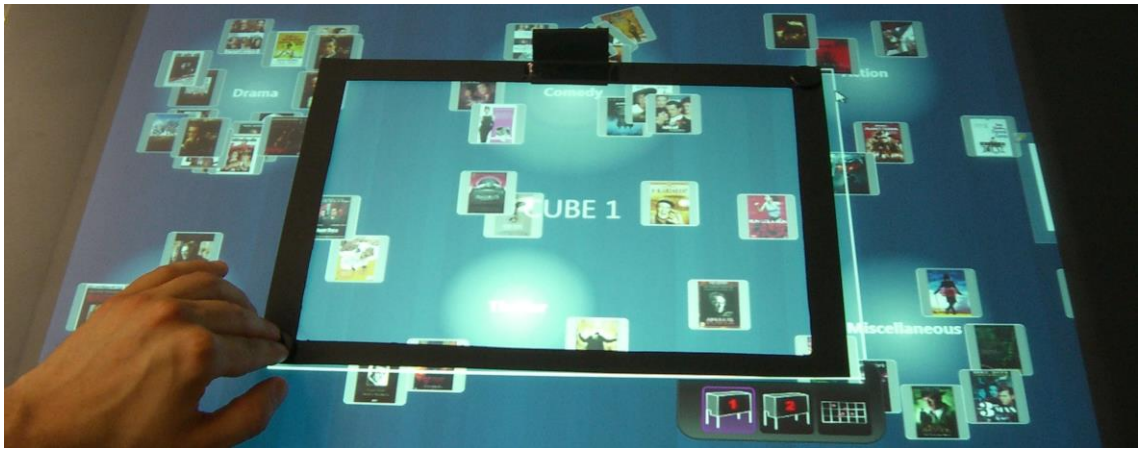


Figure 91 – A ZOIL-based prototype with a passive tangible lens on a Microsoft Surface tabletop. The mode of the lens can be selected with the virtual buttons in the lower right corner of the lens.

Following Ullmer and Ishii’s terminology, the tangible lenses in ZOIL are “passive” and not “active”. This means, that ZOIL’s lenses are not active displays or screens that are for example moved and tracked with 6 degrees of freedom in physical space to provide an augmented reality see-through display. Instead, they are passive objects without an own display that lie flat on the tabletop’s surface and can be dragged and rotated there. The lenses’ position is tracked by the vision system of the tabletop using fiducial optical markers that are sticking on the lenses’ backside. The “display” of the lens is simply the part of the tabletop’s display that lies within the frame. This creates the illusion that the lens itself serves as a special purpose display. This also enables users to interact with the content of the lens the and virtual controls surrounding it using (multi-)touch, e.g., for touching virtual buttons or zooming inside of the lens.

During the work on ZOIL and the ZOIL software framework, I explored three different lenses with different functionalities on a Microsoft Surface tabletop in the *Media Seminar Room* prototype (see section 2.1). Like in *Pad++*, The *visualization lens* could be used in different modes to show the underlying content using different information visualizations (Figure 92). Furthermore, the *Post-It lens* on the tabletop in Figure 93 creates a virtual Post-It note at its position in the information landscape. By writing on a physical piece of paper with an Anoto digital pen, all ink strokes from the paper are transmitted in real-time into its virtual counterpart. Additionally, the lens can also receive touch input for coarse-grained highlighting or drawing by finger. Finally, the *camera lens* can be used on a tabletop to control what part of the landscape becomes

visible on a second or third remote display, e.g., large vertical high-resolution displays (Figure 91). During moving the lens the view on the remote vertical displays is instantly updated in real-time so that it always shows exactly the part of the landscape that lies inside the lens' boundaries, but at a much higher resolution and detail. This creates the illusion of moving a physical camera on the "overview"-tabletop to remotely control the much higher resolution and larger "detail"-view on the remote display. As a result, a camera lens is a physical version of an overview+detail technique (*Cockburn, Karlson, and Bederson 2008*) but is distributed across two separate computers and physical displays instead of sharing a single display and machine.

Figure 92 – Different see-through visualizations (list, scatter plot, bar charts, HyperGrid) that can be displayed inside the tangible lens from Figure 91.

Figure 93 – A transparent frame with markers serves as *Post-It lens*. Left: Touch writing. Middle: Handwriting with Anoto digital pen & paper. Right: Ink strokes immediately appear on virtual Post-It.

The *AffinityTable* prototype from section 3.5.4 also uses the concept of the camera lens. However, instead of a physical cardboard frame, it uses a physical camera token that enables user to control views more flexibly. The token is put on the tabletop to remotely control the view shown on a large vertical display (Figure 94). When users slide the token on the tabletop, the content of the tabletop inside a rectangular region around the

token is continuously rendered in great detail and resolution on the vertical display. The dotted white rectangle on the tabletop in Figure 94 indicates which region is currently rendered in detail on the vertical display. However, unlike with a physical frame, its size can be manipulated by the user: A clockwise rotation of the camera token is mapped to shrinking the rectangle, i.e., zooming in. A counterclockwise rotation is mapped to enlarging the rectangle, i.e., zooming out. When enlarging the rectangle beyond the size of the tabletop, the role of both displays switch: the vertical display becomes an overview and the tabletop serves as a detail view.

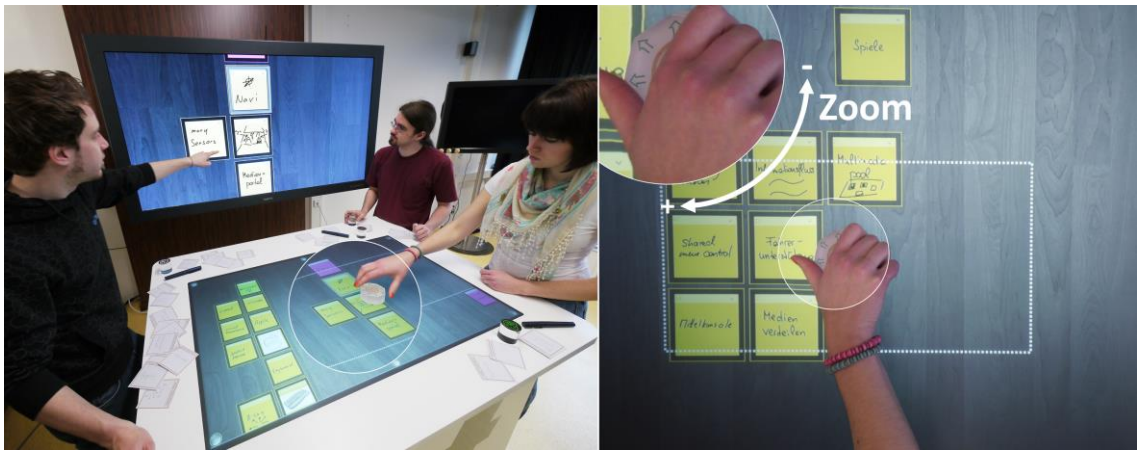


Figure 94 – The AffinityTable setup (left). By sliding and rotating the camera token on the overview-tabletop users can control the content of the vertical high-resolution 4K display. Adapted from (*Geyer, Pfeil, Höchtl et al. 2011*).

4.4 Internal ZUI Implementation in the ZOIL Software Framework

After describing the different ZUI interaction techniques that are supported by the ZOIL software framework, this section gives a brief overview of the internal implementation of ZUIs in ZOIL and the technical limitations resulting from using WPF.

Internally, WPF, and thus also the ZOIL framework, uses a hierarchical scene graph called the 'visual tree' as logical representation of visual and UI elements. The visual tree is used for rendering, clipping, applying affine transformations (e.g., translation, rotation, scaling), and also for sending, receiving, and managing UI events. In this respect, WPF follows many of the suggestions and lessons learned of ZUI toolkit pioneers such as Ken Perlin or Ben Bederson from Pad, Pad++, Jazz, and Piccolo (*Bederson, Grosjean, and Meyer 2004*). In particular, WPF supports the controlled "bubbling"' of events. If a user interacts with an element, the resulting events can travel up the visual tree to its parent control and from there to all further parent controls in the hierarchy. This enables a flexible nesting of interactive controls while managing input events at the desired hierarchical level. By this, "bubbling" enables ZOIL developers to support scenarios of 'nested user interface components' as suggested by (*Perlin and Meyer 1999*).

Furthermore, unlike popular UI toolkits such as Java Swing or WinForms, WPF natively supports a vector-based and hardware-accelerated rendering of UI controls including affine transformations. This enables ZOIL designers to directly use the full range of WPF

controls (e.g. sliders, player controls) at arbitrary scales or rotation angles without problems of pixelation (Figure 95) or the need for wrapper classes like in Jazz or Piccolo (*Bederson, Grosjean, and Meyer 2004*). As a consequence, ZOIL developers can easily integrate all kinds of WPF controls including media content such as video streams or 3D models in ZOIL's zoomable information landscape. This greatly increases the prototyping efficiency in comparison to using other hardware-accelerated graphics APIs with only rudimentary sets of user controls, e.g. OpenGL, DirectX, XNA.

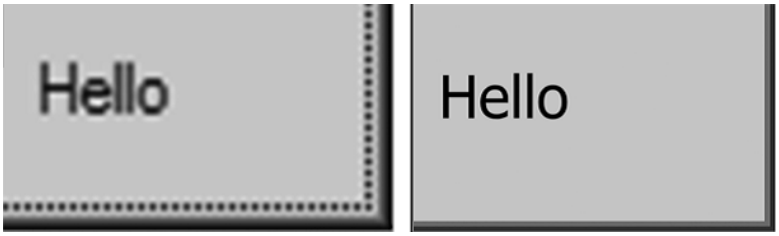


Figure 95 – Pixelation is a common problem when using non-vector based ZUI toolkits such as Jazz or Piccolo (left). WPF uses vector-based controls that can be scaled without pixelation (right). Source: (*Jetter, Zöllner, Gerken et al. 2012*).

WPF also differs from many popular commercial UI toolkits, e.g., Java Swing, WinForms, GDI, GDI+, since it uses 'retained mode rendering'[51]. This means that WPF applications do not directly cause the actual rendering of pixels and are not responsible for repainting their portion of the screen. Instead they create and update the UI's representation in the visual tree asynchronously and leave the actual execution of rendering or refreshing of screen content to WPF's rendering system. This way, ZOIL applications can offload the entire responsibility for lower level tasks such as layouting, double buffering, clipping, or occlusion culling to WPF. This also allows WPF's rendering system to optimize the rendering performance in different ways, e.g., by minimizing the amount of redrawing in the update region and making better use of the computational and memory resources of the graphics card and its GPU.

However, as discussed in several blog posts and online developer communities, the internal implementation of WPF's rendering system does not use the graphics card's GPU as efficiently as it could. For example, Morrill analyzed the performance of WPF and found that there is a greater amount of draw calls, kernel transitions, and driver load than in comparable implementations of vector graphics, e.g., in Microsoft's Direct2D API[52]. As a result, although hardware accelerated, WPF UIs have a comparably high CPU load and their frame rates quickly deteriorate with growing visual complexity. This internal problem of WPF cannot be circumvented by the ZOIL software framework or the application developer except using fewer and less complex of visual objects.

WPF's retained mode rendering system also affects the performance of ZOIL's implementation of semantic zooming. ZOIL uses a different approach than traditional

[51] More information on WPF: *http://msdn.microsoft.com/en-us/library/ms748373.aspx* (Accessed Jul 27, 2012)

[52] A Critical Deep Dive into the WPF Rendering System. *http://jeremiahmorrill.wordpress.com/2011/02/14/a-critical-deep-dive-into-the-wpf-rendering-system/* (Accessed Jul 27, 2012)

ZUI implementations. For example, ZUI objects in Pad received an 'expose event' whenever they were supposed to be rendered (*Perlin and Fox 1993*) and took care of the rendering and the semantic zooming themselves in a CPU-based 'immediate mode'-style of rendering. Using WPF, ZOIL's *ZComponent* objects (see section 4.3.5) do not receive such an event and the *ZComponentFrames* object also could not react immediately to it. Instead, the ZOIL software framework does not use an event-based model of semantic zooming but uses an asynchronous process that regularly traverses the entire visual tree (approximately once per display frame) to find all *ZComponent* objects. It then compares their actual screen size from the last rendering process to that of their different representations and, if necessary, initiates s transitions between different *ZComponentFrame* objects. Thus ZOIL's semantic zooming is executed asynchronously to WPF's rendering system (*Engl 2008: 21*).

In the different ZOIL-based demonstrators and prototypes, this approach has proved to be reliable and convenient as it follows WPF's design philosophy to offload responsibility from the application programmer to the underlying software framework. A weak point of this approach is however its additional CPU load, since a visual tree of a large canvas can easily grow to 10,000 or 100,000 nodes and has to be traversed multiple times per second. This further adds to WPF's high CPU load and, as discussed in section 3.5.5, limits the maximum amount of objects that can be displayed in the information landscape at sufficient frame rates.

Although, they were not implemented and tested in course of my PhD work, there are three promising approaches to increase the rendering performance of the ZOIL software framework in future:

- The first approach is to reduce the amount of traversing operations for ZOIL's semantic zooming by creating an index structure that manages references to all contained *ZComponent* objects instead of searching for them in the entirety of all nodes in the visual tree.
- The second approach is to reduce the complexity of the visual tree by 'virtualization' similar to the approach that is used by Kael Rowan in his WPF code sample demonstrating a zoomable canvas containing 1,000,000 items[53]. It demonstrates how large amounts of objects can be kept in the logical data model of an application, while the visual tree only contains the elements that are currently visible in the viewport.
- The third approach is to make more use of WPF's *CachedComposition* feature[54] which is currently only used in ZOIL for virtual Post-It notes containing many ink strokes. It enables caching an interactive user interface element as a bitmap to

[53] ZoomableApplication2: A Million Items. *http://blogs.msdn.com/b/kaelr/archive/2010/08/11/zoomableapplication2-a-million-items.aspx* (Accessed Jul 27, 2012).

[54] What's New for Performance in WPF in .NET 4. *http://blogs.msdn.com/b/jgoldb/archive/2010/04/12/what-s-new-for-performance-in-wpf-in-net-4.aspx* (Accessed Jul 27, 2012).

avoid full rerasterization of vector graphics during UI updates. To reduce the amount of pixelation when zooming into a cached bitmap, WPF lets programmers control at which scale the internally cached bitmap is rendered. Thus it is possible to render a high resolution bitmap before zooming animations and to use this cached bitmap instead of full rerasterization during the many visual updates that are necessary when zooming in.

A further limitation of ZOIL's ZUI implementation is based on the floating point precision of WPF's affine transformations. Internally, the ZOIL software framework applies a translation and a scale matrix on the information landscape to create the desired view of the landscape at coordinates X, Y, S (see section 4.2). Internally, WPF's transformation matrices use floating point values of the type *Double* with a double-precision 64-bit number. In informal tests, using large scale factors of 500,000 or above resulted in increased rendering artifacts due to the limited precision of *Double* values. When zooming in, objects on the screen started to randomly deviate from their expected positions leading to perceivable "jumps" with each step of zooming in or out.

While this limitation was not important for the different prototypes in this thesis, future ZOIL applications with extensive nesting of zoomable content could experience such problems and application programmers must consider new approaches to circumvent them. Unfortunately, there is currently no possibility in WPF to use the high-precision 128-bit floating-point data type *Decimal* of C# for transformations. A different approach would be to let applications create a hierarchy of several nested information landscapes. As soon as the maximum scale factor is reached during navigating a parent landscape, the application could generate a new child landscape and replace the viewport of the parent landscape with its own viewport. Ideally, these replacements are imperceptible for the user and not inhibiting the experience of continuous zooming and panning.

4.5 User Study: Post-WIMP Navigation in ZUIs

Despite the popularity of ZUIs, the body of knowledge in HCI about the effect of post-WIMP input devices on ZUI usability is surprisingly little. In particular, there are no user studies about the effect that the current shift from mouse to (multi-)touch input has on the users' spatial cognition, spatial memory and their ability to master typical ZUIs or 'panning UIs' (see section 4.2, p.113).

Studies such as (*Forlines, Wigdor, Shen et al. 2007*; *Micire, Schedlbauer, and Yanco 2007*) have documented (dis)advantages of touch such as speed and accuracy of target acquisition. However, recent findings from Embodied Cognition (*Dourish 2004*; *Gibbs 2006*; *Klemmer, Hartmann, and Takayama 2006*) provide plausible arguments why replacing the mouse with touch might also have strong effects on spatial memory and cognition that go far beyond target acquisition. For example, various studies demonstrate that the ability to transform mental images is linked to motor processes, so that rotating one's hands in the direction opposite to the required mental rotation slows down the speed of mental rotation (*Gibbs 2006*; *Wexler, Kosslyn, and Berthoz 1998*).

Others report that hand and arm gestures facilitate the maintenance of spatial representations in working memory (*Wesp, Hesse, Keutmann et al. 2001*). Therefore, it seems very plausible that an UI's input device and modality and the proprioceptive and kinesthetic feedback it provides may also have an effect on users' spatial memory. Such an effect could have a great impact on the usability of panning UIs and ZUIs where spatial memory is needed for recalling visual landmarks and their spatial relation to enable an effective and efficient navigation. For this reason, we[55] conducted two experiments to reveal if touch instead of mouse input aids users' spatial memory and improves navigation performance for such user interfaces (*Jetter, Leifert, Gerken et al. 2012*; *Schubert 2010*).

4.5.1 Study Background and Related Work

In cognitive science, the growing popularity of the view of Embodied Cognition renders the traditional Cartesian dualism with a mind-body separation as obsolete. In the novel embodied view, the body and cognitive skills coevolved from the beginning as a unit: *"(...) it is the entire system of muscles, joints, and proprioceptive and kinesthetic functions and appropriate parts of the brain that evolve and function together in a unitary way"* (*Kelso 1995*). Accordingly, recent experiments have observed strong effects that hand or arm movement has on mental tasks, images, speech or working memory (*De Ruiter 1998*; *Gibbs 2006*; *Morsella and Krauss 2004*; *Wesp, Hesse, Keutmann et al. 2001*; *Wexler, Kosslyn, and Berthoz 1998*). For example, Wesp et al. (*Wesp, Hesse, Keutmann et al. 2001*) and De Ruiter (*De Ruiter 1998*) argue that gestures may help maintain spatial images in the visuo-spatial scratchpad of Baddeley's model of working memory (*Baddeley 1986*).

Although Embodied Cognition is increasingly applied in HCI (*Dourish 2004*; *Imaz and Benyon 2007*; *Klemmer, Hartmann, and Takayama 2006*), the particular role of mouse vs. (multi-)touch for spatial cognition has not been researched yet. Traditional comparisons of input devices only focus on Fitts' law and speed/accuracy tradeoffs (*Forlines, Wigdor, Shen et al. 2007*; *MacKenzie and Buxton 1992*; *Meyer, Cohen, and Nilsen 1994*; *Micire, Schedlbauer, and Yanco 2007*). While this is an essential part of using any kind of UI, it is not subject to research here. Instead, I focus on aspects of spatial cognition and what effect the input device has on them. These aspects have a similar or even greater impact on the usability of a ZUI or panning UI than the target acquisition: They are essential for knowing *where to move* and *how to move* to desired locations in virtual space and thus they are critical for any effective and efficient usage. They demand cognitive abilities such as the *development and use of a mental spatial representation* of the virtual canvas, deciding on a path towards a currently invisible destination, and following this path

[55] My idea to observe the potential effect of touch navigation on spatial memory was a result of reading literature on embodied cognition, e.g., (*Dourish 2004*) or (*Gibbs 2006*). For this purpose, I conceived of a study design to observe this effect. This design was iteratively refined during discussions with Jens Gerken, Svenja Leifert, and Sören Schubert. The apparatus of the study was implemented by Sören Schubert according to my specifications and he used it in a first study that was published in his Master's thesis (*Schubert 2010*). Together with Svenja Leifert, I repeated the study with alterations on the apparatus and with more advanced cost functions and improved statistical analysis. The results were published in a full paper that I authored for the AVI conference (*Jetter, Leifert, Gerken et al. 2012*) and that is reproduced in the following sections.

using the input device until the destination becomes visible. While I do not know about any prior work addressing this issue, I identified four categories of related work from HCI:

Studies comparing accuracy and speed of mouse vs. touch

Meyer et al. reported that mouse outperformed touch in terms of speed for a desktop display (*Meyer, Cohen, and Nilsen 1994*). Micire et al. studied selection tasks on a tabletop and concluded that task completion was faster with touch and the error rate was comparable to the mouse for target sizes above 30mm (*Micire, Schedlbauer, and Yanco 2007*). Forlines et al. showed that users of tabletops may be better off when using a mouse for unimanual input and their fingers for bimanual input (*Forlines, Wigdor, Shen et al. 2007*). They also suggested that other design considerations such as spatial memory should play a role and further investigations into such qualities are needed.

Studies of Fitts' law for 2D tasks and zooming

The original Fitts' law models one-dimensional target acquisition as a speed vs. accuracy tradeoff. Mackenzie et al. applied this to two-dimensional tasks (*MacKenzie and Buxton 1992*) and provided the basis for above-mentioned comparative studies (*Forlines, Wigdor, Shen et al. 2007*; *Micire, Schedlbauer, and Yanco 2007*). Guiard & Beaudouin-Lafon extended Fitts' original pointing paradigm and showed that it does also apply to ZUIs (*Guiard and Beaudouin-Lafon 2004*). However, this does not help us to directly address our research questions concerning spatial memory and navigation.

Studies of touch gestures for zooming and panning

Today's dominant design is the two-finger zoom or pinching gesture (see 4.3.1). An alternative design was proposed by Malacria et al. (*Malacria, Lecolinet, and Guiard 2010*): CycloPan and CycloZoom are clutch-free single-touch gestures based on oscillatory motions. However, embodied views of spatial memory or navigation were not a part of their design rationale or evaluation. Other work on touch gestures is concerned with the design of entire multi-touch gesture sets, e.g. (*Schmidt, Nacenta, Dachselt et al. 2010*; *Wobbrock, Morris, and Wilson 2009*). However, they also were not evaluated with respect to their effect on spatial memory or navigation.

Studies of spatial memory

In the past, HCI studies of spatial memory for bookmark or window management, e.g., Data Mountain (*Robertson, Czerwinski, Larson et al. 1998*), Task Gallery (*Robertson, Dantzich, Robbins et al. 2000*), only focused on the visuo-spatial metaphors (Figure 67) on the screen and how they affect the user. The input method with mouse and keyboard was not a subject to research. More recent research on spatial and content memory completely removed user input from the study and focused solely on the effect of visual grids (*Leifert 2011*). Ebert et al. explored the combination of spatial and kinesthetic memory in front of a wall-sized 3D vision display with real depth perception (*Ebert, Deller, Steffen et al. 2009*). Users performed a memory task in a 3D scene with mouse vs. physical navigation, i.e., walking in front of the screen. However, Ebert et al.'s results

were inconclusive and cannot be applied to 2D panning UIs or ZUIs without depth perception and only hand and arm movement.

In conclusion, I am only aware of a single study in HCI that is closely related to the focus of our study and serves here as a starting point. In 2002, Tan et al. conducted a user study on the effect of kinesthetic cues on spatial memory (*Tan, Pausch, Stefanucci et al. 2002*). They compared mouse vs. touch input for a memorization task on a vertical 18.1" screen during which 28 users had to memorize objects by dragging them into given locations in an 11 by 7 invisible grid. However, in contrast to our research, the UI was neither a panning UI nor a ZUI. The grid always remained at the same absolute screen position. Remarkably, Tan et al. reported a significant 19% improvement of the accuracy of remembered locations for touch input.

4.5.2 Concepts and Definitions

For the description of this study and its results, I first define the two aspects of spatial cognition that we observed: 'spatial memory' and 'navigation'.

'Spatial memory' is an essential cognitive process that humans use to encode the space around them (*van Asselen 2005*). It develops and uses a mental representation or cognitive map in our mind (*Darken and Peterson 2001*). Different parts of the brain provide us with this ability to remember the positions and identities of objects that we have seen (*Moscovitch, Kapur, Kohler et al. 1995*). For the purpose of the study, I define spatial memory as the users' mental representation of the virtual canvas of a ZUI. It enables them to recall locations of objects based on their identity. The 'spatial memory performance' can be measured by analyzing the accuracy of the results of an explicit reproduction task. This task resembles a retrospective map drawing exercise during which users try to assign given objects to their original location in the canvas.

'Navigation' is the aggregate task of wayfinding and motoric motion (*Darken and Peterson 2001*). For the purpose of my study, I define navigation as a user activity during which a directed movement takes place from a home position in the virtual canvas to a destination object. Since the destination object is not visible from the home position, successful navigation makes use of spatial memory. Good 'navigation performance' means that the path used to move to the destination is short and ideally the shortest possible. Thus the navigation performance can be measured by calculating the spatial length of the executed navigation path. For my purposes, I consider navigation performance as a measure of length and not of time or motor costs since they would introduce noise that is not related to memory, e.g., user-specific experience with using multi-touch or individual preferences for mouse velocity or acceleration.

In the study, spatial memory performance and navigation performance were measured individually to identify potential commonalities and differences between them. For example, physical ergonomics or 'motor memory' (*Klemmer, Hartmann, and Takayama 2006*) might affect navigation performance but not spatial memory performance.

4.5.3 Description of Experiments

During the study, we conducted two experiments (E1, E2) with the hypothesis that touch instead of mouse input would result in better spatial memory performance and navigation performance. This assumption was loosely based on existing studies on the effect of hand and arm movement (*Wesp, Hesse, Keutmann et al. 2001*; *Wexler, Kosslyn, and Berthoz 1998*) but in particular on the work of Tan et al. (*Tan, Pausch, Stefanucci et al. 2002*).

Both experiments simulated real-world UIs using an abstract UI design with a spatial layout of objects that was greater than the visible screen size. E1 compared the impact of touch vs. mouse on a pure panning UI that resembled a large home screen (e.g. Windows 8) with many apps to switch between. E2 mimicked a classic ZUI such as Pad++ (*Bederson, Hollan, Perlin et al. 1996*) with semantic zooming where objects at different locations and scales reveal their details only after zooming in.

Both experiments took place at different points in time and with different participants. E1 was a part of the Master's thesis of Sören Schubert that I supervised in 2010 (*Schubert 2010*). E2 was conducted together with Svenja Leifert for our joint publication in (*Jetter, Leifert, Gerken et al. 2012*).

During the experiments participants performed navigation tasks in which frequent switching between a home position and destination objects was necessary. This kind of repeated switching and navigating between objects is typical for many real-world applications. Thereby memorization of object identities and locations becomes a by-product of normal use and makes them last in spatial memory for at least several minutes. Memorization is decisive for the UI's effectiveness, since the time for navigation and visual search is reduced drastically when object locations can be recalled from spatial memory.

Figure 96 – Physical setup of the experiments using a tabletop.

4.5.4 Experiment 1 – Panning UI

The goal of E1 was to observe whether the improved spatial memory performance with touch that Tan et al. reported can also be observed in a panning UI where the objects do not keep their absolute screen positions, but move on the screen following the users' panning operations. 20 participants (7 female, 13 male) were recruited from the campus of our university. Participants ranged in age from 18 to 28 years and were paid 7 EUR in compensation for their time.

Apparatus

For the study, a horizontal interactive tabletop (a Microsoft Surface 1) with a diameter of 30" and a resolution of 1024x768 was used. The device was used in its original "coffee-table" configuration without further elevation (Figure 96). The built-in touch tracking of the Surface was used to detect and process touch input. A wireless Logitech Anywhere Mouse MX was used for mouse input. The mouse was operated by the participants on top of a small rolling table next to the tabletop that had the same height as the Surface. To avoid occlusion of the screen with arm or mouse and to reduce physical strain, users could place the rolling table as desired.

The use of a mouse next to a tabletop is unusual, but for the purpose of the study, we had to make a trade-off between a natural use and orientation of the mouse versus keeping the users' visual frame of reference (e.g. perceived screen size, viewing angle, relative position to Surface) constant for both conditions. Since the visual frame of reference is more critical for spatial memory measurement, we decided for the latter. Also, since our study did not measure time or motor activity, but judged a device by how precisely positions are remembered after using it and how short the user's navigation paths were, this potentially unfamiliar and physically demanding mouse position could not bias our measurement too strongly. Furthermore, the decisive proprioception of relative hand movements with the mouse is not compromised by the mouse orientation on the rolling table.

The mouse sensitivity was set to the default values of the Logitech mouse driver and was kept constant in both experiments and for all participants. These values enabled participants to operate the mouse without extensive clutching while still maintaining a high level of precision and control.

At the tabletop, participants navigated a canvas using panning operations. The canvas contained a 12 by 9 grid with a spatial configuration of 18 items (Figure 97). At no time the entire grid was visible, since the screen always showed only a 4 by 3 section of the grid. The empty space in the center served as a home position without visible items. Each item was of similar size and color. The positions of the items were initially random and manually altered to avoid that participants can easily apply obvious memorization strategies, e.g., "all living things are at the bottom".

Conditions

The design of the mouse condition emulated popular panning UIs: The mouse cursor could be moved freely over the canvas. By pressing the mouse button over an arbitrary location in the canvas, the canvas could be grabbed and dragged around. In the touch condition, the same principle was applied but participants used their fingers to touch & slide for panning. In both conditions clutching was necessary for long-distance panning, since the mouse cursor or the finger could not be used outside the screen's physical boundaries.

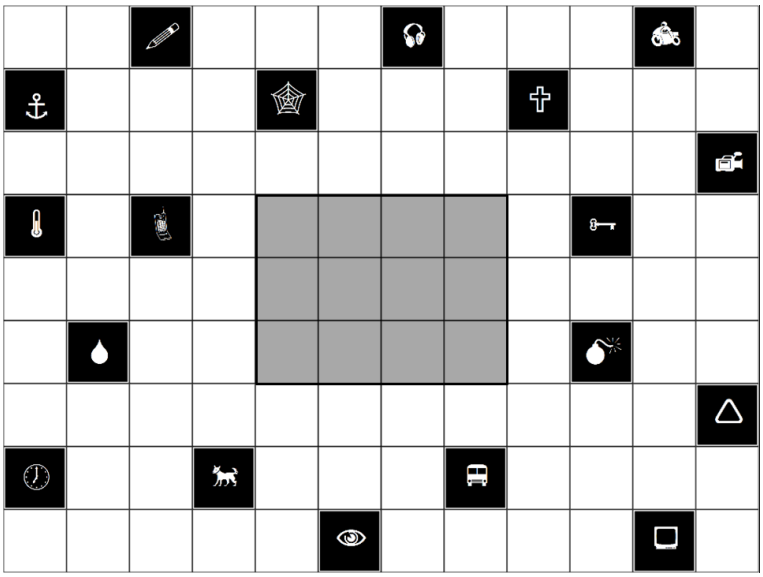


Figure 97 – A configuration from E1. The position and size of the view at the home position is highlighted in grey.

Navigation Task

In the navigation task, the system started at a home position in the empty center of the canvas where no item was visible. Then a destination item that had to be found in the canvas was shown in a transparent overlay in the center of the screen. Participants had to find this item in the canvas and pan it into the screen's center with a tolerance of 100 pixels around the center. After this, the item was considered as found and the task continued at the home position with a different item.

A sequence of 8 of the 18 items in the canvas was presented one after another as destination items. The same sequence of items was then repeated 8 times (8 blocks), resulting in 64 search trials. The first attempts in a navigation task were always bound to fail, since right after the start the configuration was still completely unknown to the participant. However, with each block, the participant saw and memorized more sections of the canvas, the items, and item locations. Therefore, we could observe the development of an increasingly accurate mental representation and increasingly efficient navigation paths.

During the navigation task, all panning operations and their X, Y, S-coordinates were continuously logged to calculate the navigation performance. We excluded the data from

the first block from data analysis to take into account the randomness of the first navigation attempts.

Spatial Memory Task / Reproduction Task

In the spatial memory task (or reproduction task), we asked participants to reproduce the learned item configuration, similar to a map drawing exercise. A random sequence of the 8 items that participants had to navigate to during the navigation task was shown and for one item after another, participants were requested to put the item into the grid at the exact location where it had been during the navigation task. Participants did not receive feedback on the accuracy of their placements and for each item of the sequence they had to start over with an empty grid. To eliminate potential influences by the unconscious use of motor or kinesthetic memory, the reconstruction task was not performed with mouse, touch or pointing at locations on the tabletop. Instead, participants used the cursor keys of a wireless keyboard to move the object inside the grid. During the spatial memory task we used the Euclidean distance as measurement and asked users to optimize for accuracy.

Procedure

We used a counterbalanced within-subject design for our experiment with input modality as the independent variable. Therefore, for each participant, the test consisted of two main parts, one for each input device. Each part was divided into three phases, in each of which one navigation task and one spatial memory task had to be completed. In phase 1, the subjects were introduced to the usage of the input device and the nature of the test tasks by a simple example and a short introduction by the experimenter. In phase 2, the practice phase, the participants were given the opportunity to get used to the input device and try out different tactics. While they still had the opportunity to ask questions, they were encouraged to constantly increase the pace during the navigation task and to achieve a high accuracy during the spatial memory task. During phase 3, the actual data collection phase, each user performed the entire navigation and reproduction task as described above. Like in the second phase, users were asked to focus on achieving a high speed respectively accuracy in the spatial memory task. Different item configurations were used in the practice and data collection phases leading to a total amount of four item configurations (two in the mouse and two in the touch condition) which were counterbalanced between the participants.

Results for Spatial Memory Performance

The comparison of the results of the spatial memory task with mouse vs. touch shows a significantly better spatial memory performance for the touch condition. While the mean error in grid units was 1.092 ($SD = 0.157$) in the mouse condition, touch led to more accurate results with a smaller mean error of 0.795 grid units ($SD = 0.096$). This difference is significant with $F_{1,19} = 5.724, p < 0.05$. Thus similar to Tan et al.'s 19% improved performance with touch for a *non-panning* UI, we also observed a 37%

improvement for a panning UI. Compared to the mouse, touch input does apparently facilitate the encoding of object locations in the users' mental representation.

Tan et al. attribute this to *"kinesthetic cues"* when using touch (*Tan, Pausch, Stefanucci et al. 2002*), but do not further elaborate on this or explain how they work. I also believe in such an effect that is based on proprioceptive and kinesthetic feedback and I provide a first hypothesis about its nature in the discussion in section 4.5.6.

In both conditions, we were surprised by the absolute spatial memory performance that exceeded our expectations. We therefore increased the difficulty of the spatial memory task in the second experiment to avoid ceiling effects.

Results for Navigation Performance

To determine the navigation performance for the conditions, we used all panning distances of the navigation task except block 1 and divided each of them by the optimal, shortest possible distance. Therefore, the closer this measure is to 1.0, the better. With touch the mean was 1.749 ($SD = 0.127$) and significantly smaller than the mean with mouse of 2.254 ($SD = 0.252$) with $F_{1,19} = 8.703,\ p < 0.05$.

As discussed before, this 29% improvement in navigation performance is not necessarily based solely on the improved spatial memory performance. Other effects could come into play here. However, since the result resonates with the increased spatial memory performance, it suggests that touch input not only results in a more accurate mental representation, but that it also enables users to effectively apply this during navigation tasks.

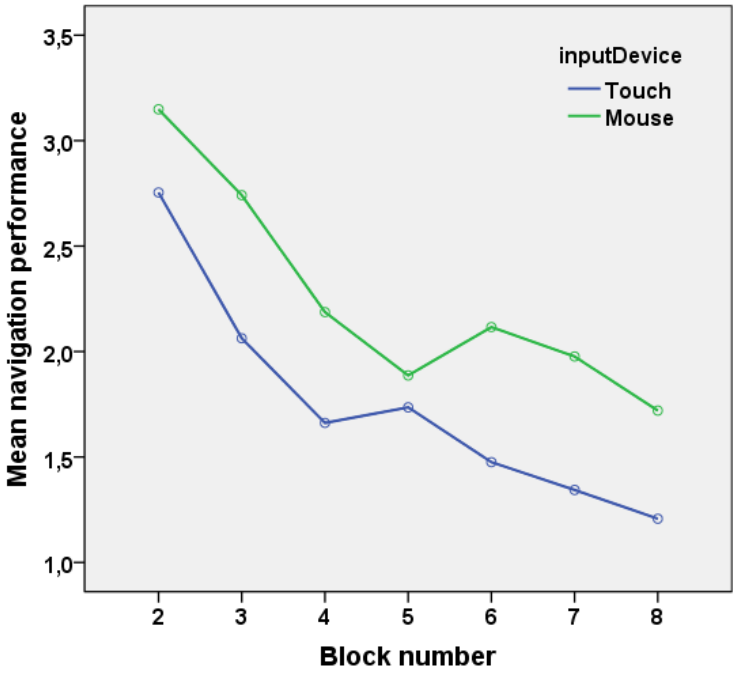


Figure 98 – Navigation performance based on mean panning distance (E1).

Regarding the development of the performance over time (Figure 98), we observed the expected learning progress that our study design was based on. Interestingly, in block 5 (touch) and block 6 (mouse) there was a sudden increase in panning distance and a decline in navigation performance. However, during the following blocks, participants returned to their previous performance and improved further. We can only speculate on the reason for this, but could imagine that this was due to a temporary decrease in attention after participants noticed the repetitive nature of the task and temporarily lost interest or motivation.

4.5.5 Experiment 2 – ZUI

E2 was very similar to E1, so that I only describe the differences between them. The most prominent difference was the use of a *zooming & panning* UI instead of a *panning* UI. Based on E1, I assumed that zooming & panning on a large horizontal touch screen would lead to better performances for spatial memory and navigation. In particular, I assumed that the employed two-finger pinching-style for zooming with multi-touch could have a strong effect, since the user cannot only specify the direction of changing S, but also the precise absolute amount by which S is changed (see 4.3.1).

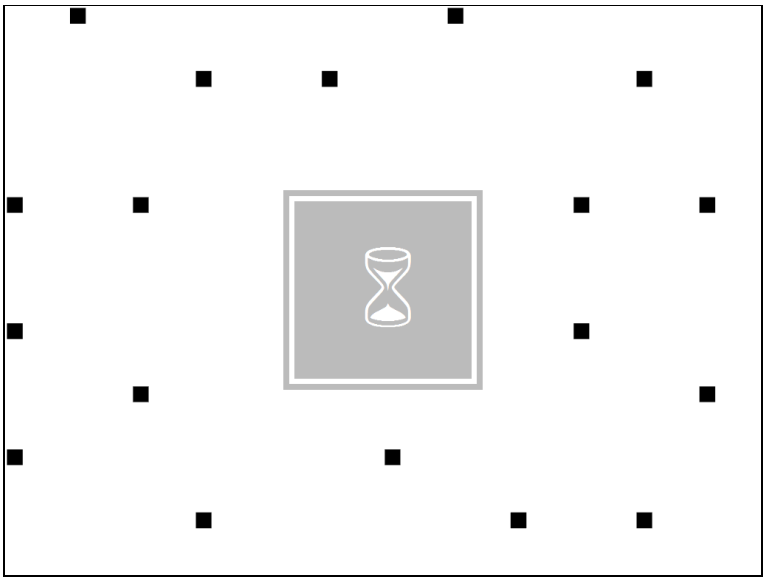


Figure 99 – Start of ZUI navigation task. The destination item to navigate to is indicated in the center. Participants can zoom & pan into the black boxes to look at the contained items.

To ensure that participants make intensive use of zooming, the initial home position was changed, so that the participants had an overview of the entire canvas from the beginning. However, the items only became visible after zooming in until S exceeded a threshold for semantic zooming. When S was below this threshold, the items' identity was not visible and only their locations in the grid were indicated by empty black boxes (Figure 99). Furthermore, the XY-distance between the items was increased compared to E1. This lead to a stronger desert fog phenomenon (see 4.3.3) and prevented users from trying to pan long distances at high scale factors without zooming out first. To avoid the potential ceiling effects for spatial memory that we discussed for the first experiment,

the width and height of the grid was quadrupled to a total of 48 by 36 grid cells. This increase in grid size did not change the nature of the tasks or the interaction, but only increased the measuring resolution of distances by factor 4 (similar to measuring a distance using feet instead of yard). Also, all grid lines or other visual landmarks except the items were removed to avoid confounding variables. Due to these alterations, participants made full use of zooming and panning as expected.

16 participants (8 female, 8 male) were recruited from the campus of our university. Participants ranged in age from 20 to 35 years and were paid 10 EUR in compensation for their time.

Results for Spatial Memory Performance

Unlike in E1, we could not observe an improvement of spatial memory performance with touch in E2. For the mouse, the mean error in grid units was 2.628 ($SD = 0.260$). For (multi-)touch, the mean error was 2.612 ($SD = 0.308$). This difference was statistically not significant ($F_{1,15} = 0.003,\ p = 0.960$).

Results for Navigation Performance

The navigation performance was determined by using the ZUI navigation cost metric introduced in section 4.2.4 to calculate the total cost of all zooming and panning operations during the navigation task (Figure 100): When applying this metric for navigation performance, the resulting mean for mouse condition is 2,988,269 ($SD = 249{,}037$) and 3,820,188 ($SD = 377{,}600$) for touch. This difference is statistically significant with $F_{1,15} = 4.727, p < 0.05$.

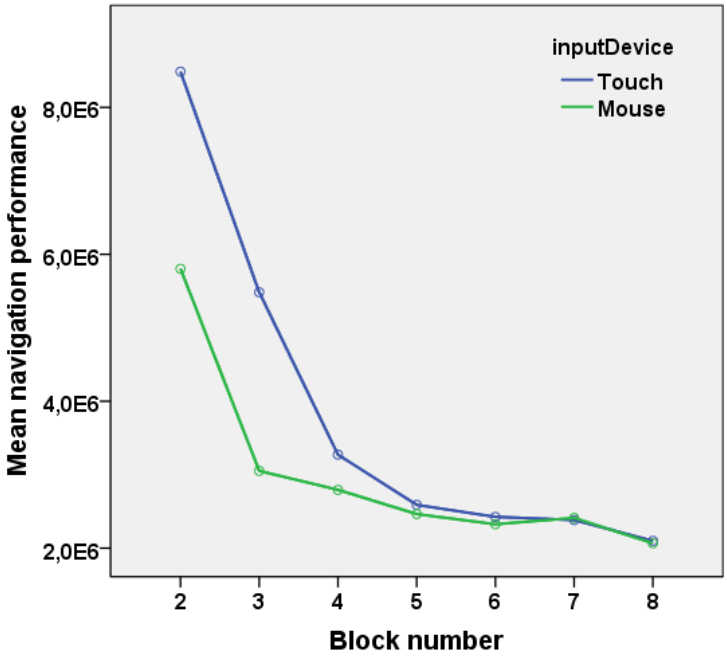


Figure 100 – Navigation performance in E2 based on the ZUI navigation cost metric.

Similar to the spatial memory performance, we could not observe the expected increase of navigation performance in the touch condition. Instead, the navigation cost in the

mouse condition was 28% smaller than with touch. This is noteworthy, since other empirical data is in favor of touch: The mean task completion time for touch was 540 s ($SD = 19.2$) and 633 s ($SD = 20.1$) for mouse, so that touch was 17% faster. The difference in the mean is statistically significant with $F_{1,15} = 15.012, p = 0.001$. Furthermore, all participants preferred touch over mouse input when asked after the experiment.

Non-Memory Effects

During the experiments, we made an observation that could explain why the navigation performance with touch did not improve and was even below that of the mouse, although participants generally preferred touch and were faster. Figure 101 visualizes this using two heat maps that show the density of interaction points for all users. The heat maps contain all screen locations where the ZRP was located during panning or zooming with mouse or touch (see 4.3.1). Using touch, participants had a tendency to focus their touches to a region below the center of the tabletop. Apparently this region was convenient to reach and touches closer to the edges were avoided, since they required greater motor activity. Also, when the target was close to the screen's edges, zooming in with two fingers seemed too cumbersome because there was not enough room for sliding two fingers apart. Most participants did not use the possibility to zoom into targets by putting one finger close to the edge and sliding only the other finger towards the center.

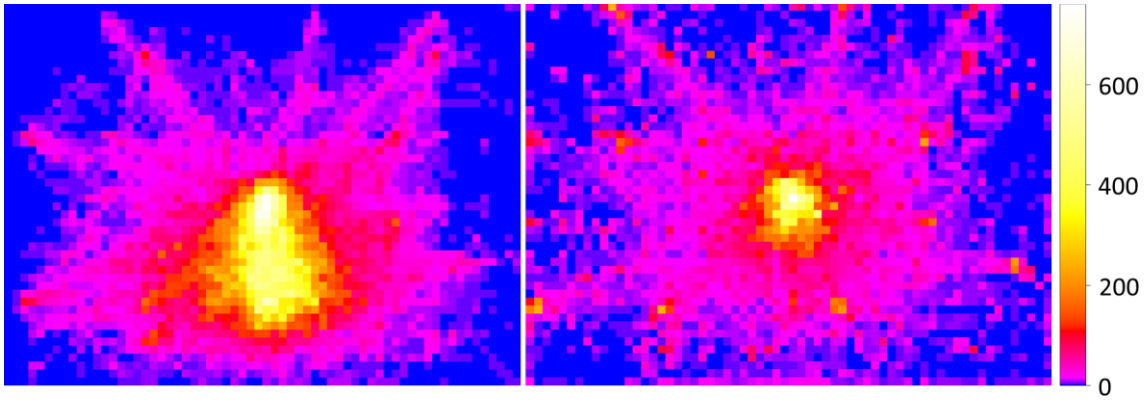


Figure 101 – Heatmaps for touch (left) and mouse (right). The color map indicates the number of logged events/region.

Using mouse, there was a similar central region where most interaction happened, but on the whole, there was a wider spread of activity across the screen. Also, participants often moved the mouse cursor against the screen's edges and used the wheel to conveniently zoom into targets along these edges. In terms of ZUI navigation cost, this approach is highly efficient and comes close to an optimal zoom-pan-trajectory. The mouse heat map in Figure 101 reveals the wider spread of interaction points across the screen and the higher density of interactions along its edges.

4.5.6 Discussion

In the following, I suggest an initial explanation of our observations that can serve as a basis for future models of the effect of the input modality on spatial memory and navigation when using (multi-)touch in a *panning* or *zooming & panning* UI. Although it has to be considered only as a first step towards a model, I believe it as an important contribution to guide further research. It also enables me to formulate first implications for design.

Touch Input for Panning UIs

Based on the results of our first experiment and previous work of Tan et al., I can conclude that there is an effect of touch vs. mouse input for non-panning and panning UIs. We assume that the greater intensity of proprioceptive and kinesthetic feedback of touch facilitates the encoding of object locations in the users' mental representation. Although the nature of E1 was different from that of Tan et al., we see commonalities in the results that we can explain as follows:

In a panning UI the absolute positions of objects on the screen constantly change with the user's panning operations. During panning, the user never sees the entire canvas but watches it through a window that travels in the canvas and only exposes temporary views on small sections. This has two consequences: First, the absolute positions on the screen are ephemeral and constantly change in time. Second, to develop a permanent mental representation of the entire canvas with global locations, users have to integrate these many ephemeral views with their temporary screen positions over time. I believe that this is not a purely visual process but can also be aided by proprioceptive and kinesthetic feedback from hand and arm movement. My assumption is that there is a greater intensity of proprioceptive and kinesthetic feedback when using touch on a tabletop instead of a mouse next to a tabletop. More precisely, I assume that the greater perceived amplitude (or length) of movements of the hand in the horizontal plane, which is sensed by stretch receptors of wrist, elbow, arm and glenohumeral joints and muscles, amplifies the encoding of locations in spatial memory. This feedback provides an additional motor perception of the panned distances that aids the process of relating the current visual input with its local positions to the existing mental representation of the canvas with its global locations. Users combine their visual perception of the panning distance in display space with the proprioceptive and kinesthetic sensation of the panning distance from motor space to form a multi-modal sense of distance. This multi-modal sense facilitates the process of integration which seems plausible, since similar effects exist for path integration and geographic orientation in 3D virtual environments (*Bakker, Werkhoven, and Passenier 1999*). Orientation based on visual flow alone proved to be more inaccurate and unreliable.

While proprioceptive and kinesthetic feedback is present for mouse and touch, there are two important differences: First, for touch screens larger than a typical mouse pad the intensity and precision of proprioceptive and kinesthetic feedback is greater. Second, on a touch screen the ratio between visual and motor space is always 1, so that the

perceived visual and motor distances always match. There is no difference between the perceived local distances and the global distances in the mental representation of the canvas. Since Tan et al.'s non-panning UI is essentially a trivial case of a panning UI, an improved spatial memory performance can be observed for both. I believe this improved spatial memory performance also facilitates navigation tasks and leads to the observed increase in navigation performance for panning UIs from experiment 1.

Touch Navigation in ZUIs

In a ZUI, not only the absolute positions of objects change on the screen, but also the visual and motor distances between them: When a user zooms in by increasing the scale factor, the visual and motor distance between two objects on the touch screen increases. When a user zooms out, the visual and motor distance decreases. Thus, unlike in a panning UI, the changing scale factors in a ZUI lead to a changing ratio between the currently visible view with its local distances and the mental representation of the canvas with its global distances. Thus the integration of the local view into the global representation has to take into account the current scale factor.

To overcome this gap during the integration process, users must rely on their visual perception. Although the pinching gestures or two-finger zoom gives users a proprioceptive and kinesthetic feedback on the current scale, I believe this feedback has not enough longevity to provide a reliable motor sense of the current scale factor. Instead, scale can only be inferred from the relative size of visual objects and landmarks, e.g., items or grids, or by consciously recalling previous zooming operations. Therefore, the benefit of a multi-modal sensation of distance cannot come into play. Accordingly, we did not observe any differences in spatial memory performance with touch vs. mouse in ZUIs.

4.5.7 Implications for Design

First, designers can expect that users benefit from using touch instead of mouse input for non-panning and panning UIs. Users are able to remember the locations of items in the canvas more precisely and they need shorter navigation paths to move to currently invisible items. We observed this for a 30" touch screen and (*Tan, Pausch, Stefanucci et al. 2002*) indicates that this also true for a 18" touch screen. However, it remains open if this also true for smaller touch screens. If the screen size comes closer to that of a typical mouse pad, this effect could diminish with the lower intensity of proprioceptive and kinesthetic feedback. A bivariate study with mouse vs. touch where screen size is introduced as a second independent variable could provide interesting insights. Ideally, such a study could also serve to better quantify the effect and contribute to the formulation of a predictive quantitative model.

Second, for ZUIs, designers cannot expect that users benefit from touch instead of mouse in terms of spatial memory and shorter navigation paths. While we witnessed significantly smaller task completion times and users unanimously preferred touch, we could not observe more accurate spatial memory or better navigation performance for

ZUIs. If designers intend to create UIs that are superior in these terms, they should consider if the amount of spatial data to provide is small enough to employ a panning UI instead.

Third, the dominant design of touch ZUIs with the pinching gesture or two-finger zoom does not come without cost. The closer the zoom target gets to the edges of the screen, the more difficulties users have with using two-finger zooming efficiently. Even after a considerable practice phase, our participants mostly preferred to pan the target into the center before zooming in. In terms of the length of ZUI navigation paths this is inefficient and the mouse performs significantly better. A potential solution could be ZUIs where the touch-sensitive area is greater than the viewport. Such a touch-sensitive rim could possibly afford more efficient two-finger zooming at the viewport's edges. It could also be worthwhile to search for new solutions for the problem of losing track of scale during zooming by introducing better visual cues or landmarks.

4.5.8 Conclusion of the Study & Future Work

In the previous sections, I have presented results from two experiments that enabled us to observe whether multi-touch instead of mouse input improves users' spatial memory and navigation performance in panning UIs and ZUIs. For panning UIs with touch input, we observed a 37% increase of spatial memory performance and a 29% increase in navigation performance. For ZUIs, we observed a unanimous user preference and a 17% improvement in task completion times for touch, but we could not observe better spatial memory and navigation performance. I provided an explanation for this effect based on a multi-modal sensation of distance. While panning UIs can benefit from this sensation, this is not the case for ZUIs due to their changing scale factors.

We also observed a 28% better navigation performance in terms of path lengths in a ZUI with mouse. We can attribute this to frequent problems that participants had with two-finger or pinching gestures for zooming into targets that were close to the screen's edges. Based on these results, I suggested first implications for design to guide interaction designers.

As future work, I suggest bi-variate studies with mouse vs. touch and different screen sizes as independent variables to contribute to the formulation of a predictive quantitative model of the observed effects. In particular, further research on smaller screen sizes or touch pads would help to validate our findings. A follow-up study could also help to shed light on the different nature of the curves for navigation performance. For the panning UI (Figure 98), the mouse curve and the touch curve seem to be parallel with a fairly constant difference between them. For the ZUI (Figure 100), the mouse and touch curve seem to have an asymptotic behavior and meet in block 7 and 8. A follow-up study should increase the number of blocks for the panning UI to observe if such an asymptotic behavior becomes observable after more than 8 blocks.

More generally, I would also like to motivate other researchers to conduct studies to understand the effects of Embodied Cognition on HCI. Cognitive science provides strong

evidence for the effects that body movement inevitably has on cognitive functions such as memory but also language use, social behavior and emotional attachment to other people or things (*Gibbs 2006*). HCI could strongly benefit from understanding and exploiting these effects when designing future user interfaces and gesture sets for gestural and touch interaction.

4.6 P#2: “Provide a Zoomable User Interface for Navigation with Semantic Zooming.”

As a summary of this chapter, I formulate the *2nd ZOIL design principle* about the role of ZUIs for ZOIL as follows:

“Provide a Zoomable User Interface for navigation with semantic zooming.”

This includes providing following interaction and visualization techniques and considering following interaction designs and design tradeoffs:

1.) ZOIL introduces a spatially continuous zoomable workspace (the ‘information landscape’) as a model-world UI for natural and consistent management of digital information items and interaction with virtual tools. Instead of a file system with folders and file names, ZOIL organizes information and tools in the landscape’s spatial structure that is not a part of the interface but a part of the content.

3.) By using a zoomable information landscape, ZOIL avoids using overly concrete real-world metaphors while still exploiting users’ familiarity with fundamental concepts and mechanisms from real-world experience, e.g., movement in space and the persistence of objects. Thereby ZOIL taps into our natural spatial and geographic ways of thinking.

4.) Animated zooming can dramatically reduce the users’ cognitive load, but is also easy to do badly. Therefore, fine-tuning of pan-zoom trajectories and animation durations is important. Furthermore, non-linear space-time functions can create a sensation of a more natural and physical motion.

5.) Mouse and multi-touch manipulations for zooming and panning should allow combining zooming and panning operations without mode switching and users should be able to freely control the zooming reference point. For multi-touch ZUIs the pinching gesture (or two-finger zoom) has the further advantage that it also allows rotation and an absolute control of the scale factor.

6.) Desert fog situations should be avoided by providing techniques such as overview+detail, parallax zooming, or restricting users to animated zooms between designated zoom targets.

7.) Use semantic zooming for smooth transitions between iconic representations and full-text/full-functionality to enable an object-oriented interaction that reduces the amount of computer administrative debris and the cognitive load of managing stacked windows and details-on-demand.

8.) Provide (tangible) lenses with meta-data visualization tools for browsing, searching, and filtering in large collections of objects. Use virtual lenses to persist results of filtering operations in the information landscape. Use tangible lenses to control overview+detail views across device boundaries or to create and edit virtual objects with digital pen and paper.

9.) Provide users with portals or multi-focus ZUIs to let them create or destroy multiple simultaneous views. This enables them to focus on different locations and scales in the information landscape at the same time and thus facilitates comparing objects or moving objects between distant regions.

10.) For panning-only UIs, (multi-)touch input makes users remember the object locations more precisely and they need shorter navigation paths. For ZUIs, multi-touch does not improve spatial memory or shorten navigation paths, but is superior in task completion times and user preference.

11.) In ZUIs, the multi-touch pinching gesture or two-finger zoom causes problems when trying to zoom into objects that are close to the edges of a touch-enabled screen. Therefore, when using multi-touch instead of mouse for a ZUI, only a smaller region of the screen is used for interaction and the amount of panning into this center is increased.

5 Space to Think, Annotate, and Collaborate

This chapter introduces and discusses the key role that virtual and physical space play for the design of ZOIL-based user interfaces. Based on empirical studies from HCI and cognitive science (*Andrews, Endert, and North 2010*; *Kidd 1994*; *Kirsh 1995*), this chapter argues for considering space an integral part of human cognition and a key resource for collaborative knowledge work. Building upon the role of virtual space in ZUIs from the previous chapter, this chapter explains how virtual and physical space can be used in a ZOIL user interface for facilitating *sensemaking*, enabling *marks* and *annotations*, and coordinating *mixed-focus collaboration*.

In the first part of this chapter, ZOIL's information landscape is used as a virtual space for incrementally arranging, resizing, rotating, and clustering objects in space and scale for the purpose of *sensemaking* (*Andrews, Endert, and North 2010*). Virtual space is also used to enable *marks* and *annotations* that can serve as a secondary notation during knowledge work to record things that have not been anticipated when designing a system and its primary visual notation (*Blackwell and Green 2003*; *Kidd 1994*). After discussing the underlying foundations and insights from previous and related work, different uses of space in ZOIL example prototypes are illustrated and results of a user study of the *DeskPiles* prototype are presented that reveal how space and annotation is used in realistic usage situations.

The second part of this chapter discusses the role of space for collaboration. This role is introduced by first looking at how users partition physical space into territories to coordinate their collaboration around traditional tables (*S. D. Scott, Carpendale, and Inkpen 2004*) and interactive tabletops (*Jetter, Gerken, Zöllner et al. 2011*) during "*mixed-focus collaboration*" (*Tang, Tory, Po et al. 2006*). Then this is applied to the domain of collaboration in non-physical virtual workspaces, more concretely in ZOIL's zoomable information landscape. In ZOIL, multiple active input/output devices serve as cameras into the shared information landscape. Each camera can be individually controlled to view different, same, or overlapping regions of the workspace in order to support many different coupling styles and fluid transitions between tightly-coupled and loosely-coupled collaboration.

Based on these discussions of previous work, design examples, and empirical observations, this chapter formulates the *3rd* and *4th ZOIL design principle*.

This chapter uses figures from the *DeskPiles* prototype and paragraphs about ZOIL's camera concept from a journal article on the ZOIL paradigm (*Jetter, Zöllner, Gerken et al. 2012*).

5.1 Space as a Cognitive Resource

The previous chapter used a zoomable information landscape to organize a system's information items and functionality in space and scale. The main benefit of spatiality was to enable users to use their innate and learned spatial skills for more efficient memorization and navigation. However, according to work such as (*Andrews, Endert, and North 2010*; *Hollan, Hutchins, and Kirsh 2000*; *Kidd 1994*; *Kirsh 1995*), space is much more than just a navigational cue or a simple extension of memory. The following sections elaborate on the many uses of space during cognitively demanding tasks such as individual or collaborative knowledge work.

5.1.1 Intelligent Use of Space

Following Kirsh's account of the *intelligent use of space* (*Kirsh 1995*), space must be considered an important cognitive resource that is constantly managed by humans, much like time, memory, and energy: *"In having a body, we are spatially located creatures: we must always be facing some direction, have only certain objects in view, be within reach of certain others. How we manage space around us, then, is not an afterthought; it is an integral part of the way we think, plan and behave, a central element in the way we shape the very world that constrains and guides our behavior"* (*Kirsh 1995*).

Kirsh observed how managing space enables us to bring the time and memory demands of our tasks down to workable levels and to increase the reliability of execution and the number of jobs we can handle at once. He identifies three main categories for functions of space: spatial arrangements that *simplify choice*, spatial arrangements that *simplify perception*, and spatial dynamics that *simplify internal computation*.

Spatial Arrangements that Simplify Choice

An example for using space to *simplify choice* are *"production lines"* (*Kirsh 1995*). We often use these lines to spatially decompose a complex real-world task (e.g., preparing a salad in a kitchen) into functional stations where specific subtasks are performed at different locations. For example, all vegetables for a salad are placed beside the sink. As each vegetable is washed, they are placed aside, separating them from the unwashed vegetables. In a next step, the washed vegetables are transferred to beside the cutting board, where knives are kept and the vegetables can be chopped and put into a bowl. In Kirsh's salad example, space is used to simplify choice by creating stations that effectively trigger an *"action frame"* or *"task context"* (*Kirsh 1995*) in which only a fraction of all possible actions have to be considered. The local affordances are defined by equipment such as knives, sink, bowl, etc. and make clear what can and must be done. *"The virtue of spatially decomposing the task is that one need not consult a plan, except at the very highest level, to know that to do"* (*Kirsh 1995*).

While Kirsh's observations were made in the non-digital physical world, they are also relevant for the design of user interfaces. For example, application windows typically contain pull-down menus or tools that are spatially close to the content to manipulate and thus ideally trigger a "task context" in the sense of Kirsh. The users' window

management often resembles the creation of "production lines", for example when arranging multiple windows side-by-side to spatially decompose a task that involves multiple applications. Similarly, in a ZOIL user interface, semantic zooming reveals the associated functionality of an object – and thus the local affordances – inside of the object itself. ZOIL's information landscape enables users to create spatial arrangements of these objects, e.g., multiple objects in a straight line to create digital equivalents of "production lines".

Spatial Arrangements that Simplify Perception

Kirsh uses the example of segregating washed and unwashed vegetables while preparing a salad to explain the use of spatial arrangements that *simplify perception*. For example, clean and dirty tomatoes can be hard to tell apart, so that they are put in different places after washing them to highlight their differences, e.g., from the left to the right of the sink. Such clustering also helps because it is harder to lose big things (a cluster of tomatos) than little ones (a single tomato) and to miss seeing what a whole group is good for. Perception can also be simplified by using space for symbolic positioning or marking of an object, for example by putting a knife on a measured piece of butter to mark it as the piece to use and to segregate it from the unmeasured rest (*Kirsh 1995*).

User interfaces like the *Media Seminar Room* (see section 2.1, Figure 11, p. 31) make intense use of clustering. An initial clustering (e.g., by genre) can segregate objects of one kind from others. Furthermore, users can create own clusters with selections of objects, e.g., particular relevant objects that are put aside. The *DeskPiles* prototype additionally explores symbolic positioning and a range of marking and annotation techniques that are discussed in greater detail in the user study below (see section 5.7, p.170).

Spatial Dynamics to Save Internal Computation

Kirsh's example for *spatial dynamics to save internal computation* is that of striking a bundle of spaghetti on a table to single out the tallest noodle from its neighbors. Unlike sorting algorithms, this performs an instant sort computation by using the material and spatial properties of the world. *"The method does not involve symbolic computation, or an item-by-item comparison. It works by creating a visual cue that serves to make the property in question explicit"* (*Kirsh 1995*). Another example that Kirsh mentions is taken from laboratory studies of users playing fast-paced computer games. In the case of the Tetris game, players display a burst of rotations after a new piece enters the screen to rapidly generate the mental icons they need to try out candidate placements. Rotating a piece externally in the space of the screen takes far less time than internally rotating its mental image: *"By shifting the burden of imagery formation away from the mental rotation component, and giving it, instead, to the faster working motor and perceptual systems, an agent can save hundreds of milliseconds. This is what is meant by saying the agent uses the world to save internal computation"* (*Kirsh 1995*).

Conclusion

Kirsh concludes his account of the intelligent use of space by criticizing that theorists have dwelled on the intelligent use of time, hardly considering space, although vision, our primary spatial sense, is one of humanity's most powerful capacities. He argues that much of our everyday competence might come from informationally structuring our workspaces and thus space is an integral but underrated resource in human cognition. Kirsh's examples from the non-digital and digital world (e.g., kitchen, Tetris) hint at the importance of actively considering space, not only when designing physical work environments, but also when designing virtual work environments and interactive systems for cognitively demanding tasks such as collaborative knowledge work.

5.1.2 Space in Distributed Cognition

Theories of *distributed cognition* consider human cognitive processes and the involved knowledge, memories, and facts as *distributed* across the members of a social group, across mental, material, or environmental structure, and through time (*Hutchins 2001*). As opposed to mainstream cognitive science, distributed cognition looks for a broad class of cognitive events "in the wild" and does not expect all such events to be encompassed by the skin or skull of an individual (*Hutchins 2001*). In this view, cognition is not confined to an individual human agent. It must be understood by looking at the entire sociotechnical system (e.g., a ship's brigde, an airline cockpit), including all other agents, their communication, their tools, and their environment.

For example, in their work on distributed cognition for HCI, Hollan et al. observed that the environment provides more than simply additional memory but also opportunities to make use of a different set of internal and external processes for achieving the users' tasks (*Hollan, Hutchins, and Kirsh 2000*). Hollan et al. also refer to Kirsh's *intelligent use of space* and as a part of this research agenda, they suggest to observe how people manipulate virtual space and objects within a multiscale ZUI, e.g., Pad++ (*Bederson, Hollan, Perlin et al. 1996*), because ZUIs enable users to establish a tight coupling of interface components and cognition: *"(...) it is apparent that how users manipulate icons, objects, and emergent structure is not incidental to their cognition; it is part of their thinking process, part of the distributed process of achieving cognitive goals. They leave certain portals open to remind them of potentially useful information or to keep changes nicely visualized; they shift objects in size to emphasize their relative importance; and they move collections of things in and out of their primary workspace when they want to keep certain information around but have other concerns that are more pressing"* (*Hollan, Hutchins, and Kirsh 2000*). In the following sections, such benefits of space and the use of space in ZUIs are discussed for and analyzed during collaborative knowledge work in ZOIL-based interactive spaces. This discussion and analysis is structured into three functions of space: facilitating *sensemaking* (section 5.2, p.161 and 5.4, p.164), enabling *marks* and *annotations* (section 5.5, p.167), and coordinating *mixed-focus collaboration* (section 0, p.186).

5.2 Sensemaking – Using Space to Think

In their work, Andrews et al. focus on how increased physical display space (Figure 102) affects the cognitively demanding task of *sensemaking* (*Andrews, Endert, and North 2010*). Based on (*Card 2008*; *Pirolli and Card 2005*), they define *sensemaking* as the process of building understanding out of a collection of data. They give examples of simple or regular activities that involve sensemaking like choosing a new phone or doing task management to more critical problems such as deconstructing what happened to the market or discovering a terror plot based on intelligence data. As such, sensemaking is an integral part of *knowledge work* and in particular of Card's *knowledge crystallization operators* (see section 1.5.4, p.17).

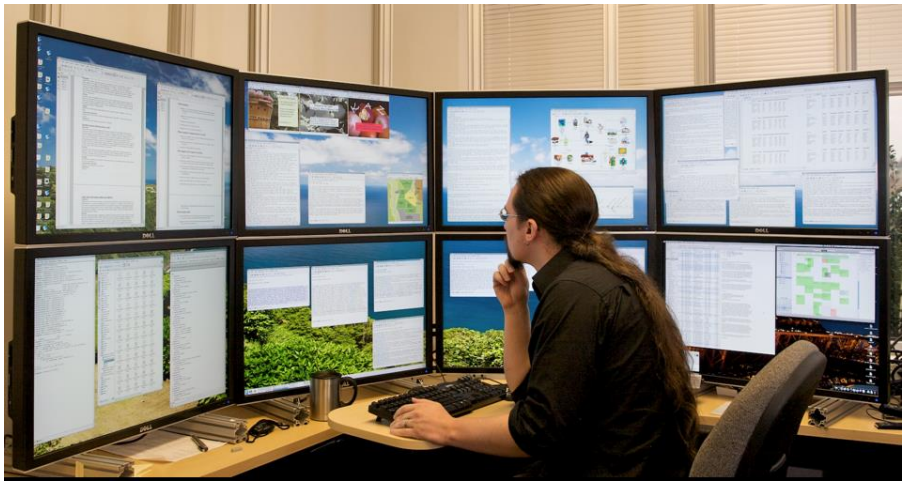


Figure 102 – The "analyst's workstation" with a 4x2 grid of 30" LCD panels with a total resolution of 10,240 x 3,200 pixels. Source: (*Andrews, Endert, and North 2010*).

Andrews et al. consider sensemaking a fundamentally human activity. Technology can provide support for searching, filtering, isolating, or visualizing, but it cannot provide understanding. This resonates with Kidd's (*Kidd 1994*) notion of knowledge work and that software for knowledge workers should avoid trying to *"understand"* the information it is holding or trying to predict what the user wants to do with it. True knowledge work cannot be automated (see section 1.5, p.14).

Andrews et al. and Kidd also agree on the importance of space and spatially laying out physical or virtual information items during sensemaking and knowledge work:

Kidd observed that physical space, such as desks or floors, can serve as a temporary holding pattern for paper-based inputs and ideas which cannot be categorized and for which there is no decision how to use them yet. At this stage, filing is uncomfortable, because users cannot reliably say when they will want to use a particular piece of information or to which of their future outputs it will relate (*Kidd 1994*). Kidd is also critical about the small displays of mobile devices since they force users to classify virtual items immediately when they are received or generated. If knowledge workers jot down an idea on such devices, they will be forced to classify the inherently unclassifiable and it is unlikely that the idea informs them later as it will have

disappeared for ever into the bowels of the device (*Kidd 1994*). As Kidd observed, users of physical paper are not forced to file or classify immediately. Furthermore, the layout of physical materials on their desk gives them powerful and immediate contextual cues to recover a complex set of threads without difficulty and delay after interruptions. Thus using physical space to create spatial layouts and piles of papers also serves as a representation of the users' context and activities.

Andrews et al. observe the same for virtual items on large displays: Users establish a *"work zone"* that serves as the primary focus and clusters or piles of *"potentially interesting"* items that cannot be categorized or filed yet. In their studies, these zones, clusters, or piles were used as alternatives to previous techniques, such as filing documents individually into directories or printing them out and organizing the papers physically, either spread out on a table or into piles on the desk (*Andrews, Endert, and North 2010*).

Andrews et al. and Kidd also agree on the importance of the use of space for reflection, reasoning, and experimentation:

Kidd observed how users use pieces and piles of papers and their spatial layout on desks, floors, or walls as a primitive language to reflect on the problem they are working on by moving papers around, marking them, and establishing orders. *"(...) a knowledge worker needs structures which are both flexible in their semantics and generative in nature. There is evidence that spatial relationships are powerful in this regard at a pre-linguistic stage of reasoning, ie. objects and their spatial relations provide a primitive means for creating, exploring and changing structures which can inform us in novel ways"* (*Kidd 1994*).

In their studies of sensemaking on large displays, Andrews et al. also observed similar uses of display space for representation and reflection. They observed that space provides a flexible "*semantic layer*" that adds meaning to displayed information: *"There are many relationships that can be represented spatially, such as ordering, proximity, and alignment (...). Space can also serve as a medium for creating complex structures like clusters, lists, and heterogeneous interrelated types (...). Using space in this fashion reduces the need for elaborate internal models by replacing memorization and computation with perception. It is, for example, far easier to arrange objects in space and use the perceptual system to recognize categories and properties present in a collection of objects than it is to try to memorize all of the characteristics of every object and internally compute relationships"* (*Andrews, Endert, and North 2010*).

A further advantage of space that Andrews et al. discuss is its great flexibility that enables experimental exploration. Unless explicitly specified (e.g., with a coordinate axis), space imposes no strict interpretation. The meaning of space and organizations can evolve with understanding, so that at the start of the sensemaking process, when nothing is known about the data, there is no need for a premature choice of an inappropriate and restrictive organizational structure. By starting to work in a freeform and loosely structured environment, users can experiment, make rough categorizations, or build *ad hoc* structures that are developed further when understanding starts to grow and with it,

a better sense of how to organize the data (*Andrews, Endert, and North 2010*). This resonates with Kidd's demand for structures which are both flexible in their semantics and generative in nature.

In the context of spatial hypertext, Shipman refers to this use of space as *incremental formalism* (*Frank M. Shipman, Hsieh, Maloor et al. 2001*). In ZOIL, the flexible use of space for *incremental formalisms* is intended to support early phases of sensemaking during which precise visual formalisms or information visualization techniques would be inappropriate and too restricting. For example, Andrews et al. observed how a user gradually transformed an initially loose cluster of a growing number of aligned documents into a layout similar to a chronological timeline. However, this timeline was intended as an intermediate artifact of sensemaking and not as a precise visualization for presentation or discussion with others. This became evident, when at some stage, the documents were coincidentally already in order, albeit in reverse. Rather than reorder them, the user continued to build and use the timeline backwards from then on as this was sufficient for this user's personal interpretation and understanding.

5.3 Physical vs. Virtual Space

In their work, Kirsh and Kidd considered *physical space* on desks and floors. Furthermore, Andrews et al. used many large screens to create a single large physical display that simulated the space, size, and the resolution of physical whiteboards including their benefit of physical navigation, e.g., by head or eye movement (Figure 102, p.161). While large high-resolution screens can also be used in a ZOIL interactive space, e.g., in *AffinityTable* (Figure 94, p.136), ZOIL typically provides space in zoomable user interfaces that is *virtual* and not *physical*. It is a largely unexplored question to what extent the benefits of space for sensemaking from physical space translate to this virtual space. Therefore, Andrews et al. suggest exploring the tradeoffs between the virtual space of ZUIs and physical space of large displays in user studies (*Andrews, Endert, and North 2010*). In a first step, following theoretical considerations could help to shed light on the question whether these benefits exist for physical *and* virtual space:

For Andrews et al., one of the advantages of large, high-resolution displays is that objects can be arranged spatially at meaningful positions and, at the same time, viewed at a detailed level (*Andrews, Endert, and North 2010*). Unlike a small low-resolution screen, the size of the display and its resolution enable users to arrange objects in spatially meaningful ways while retaining their detailed representations. There is no need to strictly separate between the small representations for storage and management purposes (e.g., thumbnails, icons, or labels) and the detail views of the actual objects in windows (e.g., documents, web pages, images, or presentations). Using semantic zooming, ZOIL achieves a similar effect. The objects of interest can be arranged in meaningful ways in the information landscape and their details can be accessed on-the-spot by zooming in. There is no spatial separation between small representations and detail views and moving between both is a visually and cognitively continuous process.

A further advantage that Andrews et al. discuss for large physical displays is that objects can be placed in a persistent location in space, allowing the user to navigate using physical navigation rather than having to switch between sensemaking tasks (e.g., reading, annotating, search) and view management (e.g., zooming, panning) (*Andrews, Endert, and North 2010*). Although ZOIL's information landscape is persistent, this does not circumvent the need for ZUI navigation and view management. However, as discussed in section 4.3.1 (p.119), the cognitive load for view management in a ZUI is reduced by introducing combined modeless zooming & panning with multi-touch gestures, animated zooming into zoom targets, and avoiding the occluding windows of WIMP interfaces. Therefore, it appears plausible that ZUIs also benefit to some degree from persistent location in virtual space, even if not to the same extent as the persistent locations in physical space of Andrews et al.'s large, high-resolution displays.

5.4 Space for Sensemaking in ZOIL

The ZOIL example prototypes explore different approaches for providing space for sensemaking, clustering, and incremental formalisms. They range from free positioning, rotating, and resizing of objects on a zoomable canvas with multi-touch (see *Media Seminar Room*, section 2.1, p.30) or mechanisms such as snapping objects into grid cells in *DeskPiles* (see section 2.2, p.37) to create and move clusters using inter-object magnetism in *AffinityTable* (see section 3.5.4, Figure 60, p.92). The following section introduces the different approaches to demonstrate the possible design space for manipulating space in a ZOIL user interface.

Figure 103 – An example landscape with free clustering of items in space and scale.

5.4.1 Media Seminar Room

Figure 103 of the *Media Seminar Room* shows how objects can be positioned freely in ZOIL's zoomable information landscape at arbitrary positions, sizes, and angles. The users' interaction with objects is similar to that of popular tabletop applications for photo browsing and sharing, e.g., when using the "ScatterView" control[56] of the Microsoft Surface SDK. When using multi-touch gestures, ZOIL's information landscape resembles a "ScatterView" that, unlike the original, is a fully zoomable user interface and whose background can be zoomed and panned using pinching and sliding gestures. Furthermore, contained objects are zoomed semantically and not only geometrically.

5.4.2 DeskPiles

While entirely free placing in space and scale maximizes the users' freedom during sensemaking, it also increases their effort for manually aligning objects, establishing regular layouts, and giving objects identical sizes. In practice, objects often overlap and occlude each other. During informal evaluations of the *Media Seminar Room*, it became obvious that entirely free placing can become demanding and time-consuming, in particular, when many objects have to be aligned or when they become very large or very small after accidental multi-touch manipulations. Establishing regular visual and spatial structures is not just a matter of aesthetics, but is important for efficient visual processing of information by users, for example in the light of the Gestalt laws of proximity, similarity, or continuity (*Ware 2004*).

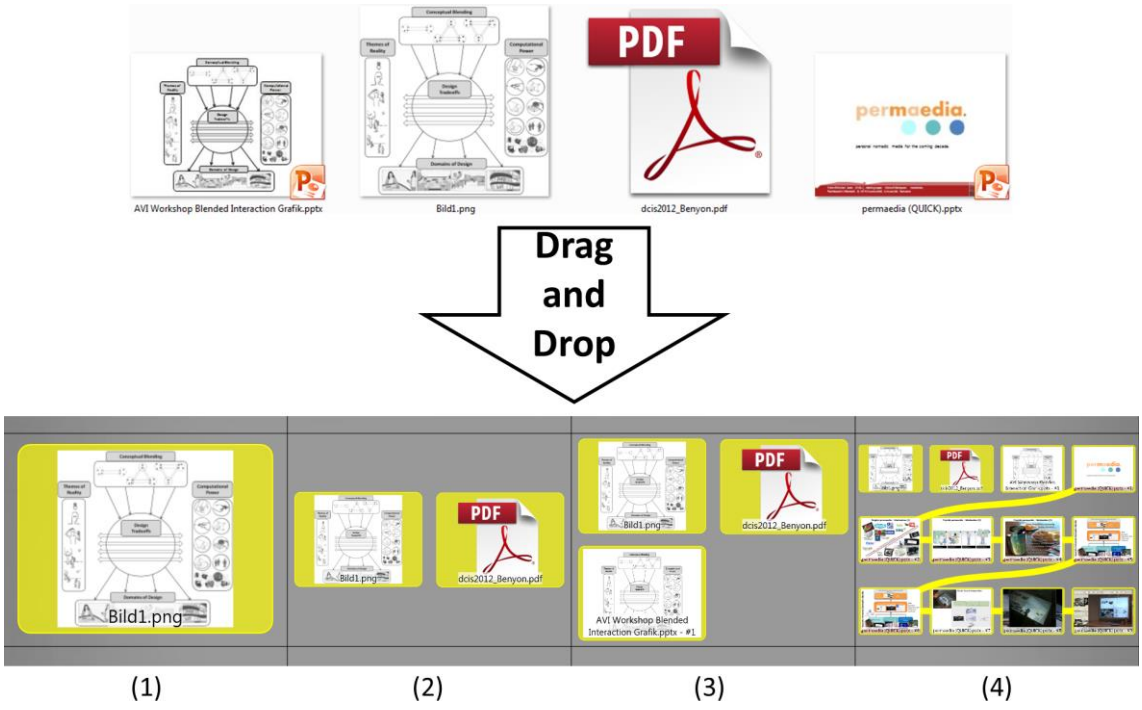


Figure 104 – Populating a cell with a growing number of objects in *DeskPiles*. From left to right: (1) single image object, (2) image & PDF, (3) image, PDF & single slide, (4) image, PDF, single slide & slide deck.

[56] ScatterView Overview: *http://msdn.microsoft.com/en-us/library/ee804791(v=surface.10).aspx* (Accessed Sep 4, 2012)

For this reason, the *DeskPiles* prototype (see section 2.2, p. 37) introduces an alternative concept of providing space in a ZUI that restricts object locations, orientations, and sizes. It uses a multi-scale variant of a "snap-to-grid" mechanism. By dragging an information item in an empty cell, the item is scaled to the cell size. If further items are dragged into the cell, all items inside the cell are automatically rescaled and repositioned, so that they appear in a regular grid layout (Figure 104). Thus the grid cells support users in arranging and aligning items to create regular layouts within visible geometric boundaries. As soon as users run out of cells, new cells can be created around the existing ones. As already described in section 2.2 (p.37), the grid can still be navigated freely by continuous zooming and panning and all items can be explored individually by semantic zooming. In the user study that is described in section 5.7 (p.170), users' feelings towards this grid approach were mixed. Participants felt it helped arranging items in initial phases, others found it constraining and wished to arrange items in a more *"organic"* and *"freeform"* space.

5.4.3 AffinityTable

In subsequent work, that is not part of my PhD project, Geyer et al. created a further approach to facilitate clustering in ZOIL's information landscape that is a tradeoff between the freeform approach of the *Media Seminar Room* and the more restrictive grid in *DeskPiles*. In this thesis, this approach is included for the reason of completeness and for illustrating the possible design space for manipulating space in ZOIL user interfaces.

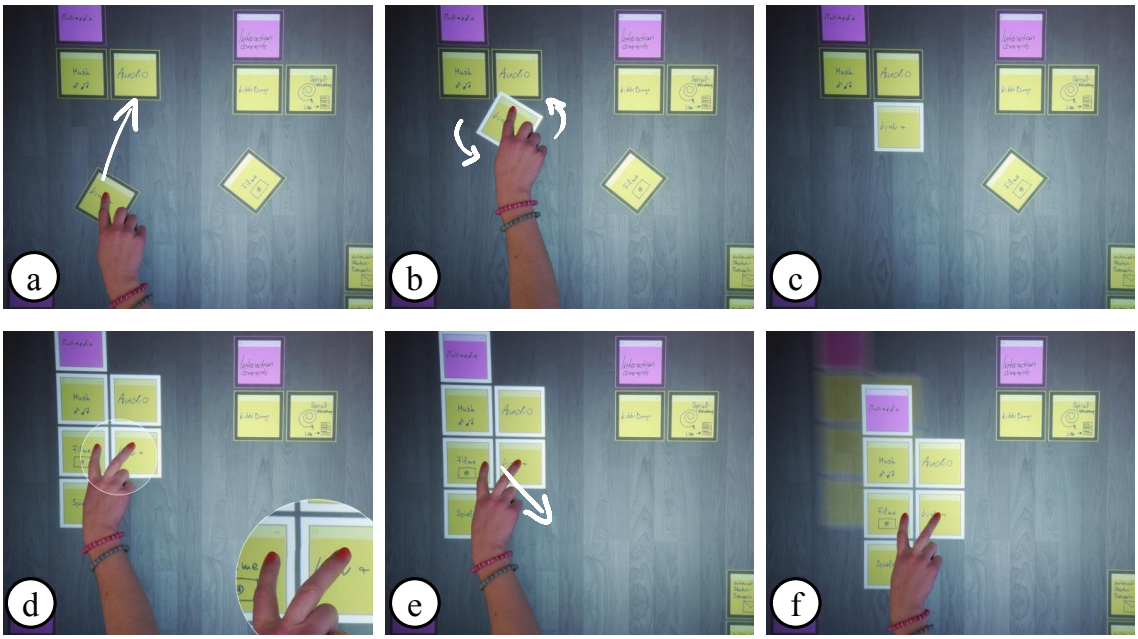


Figure 105 – Interaction techniques for clustering notes in the ZOIL-based AffinityTable prototype. Source: (*Geyer, Pfeil, Höchtl et al. 2011*).

In the *AffinityTable* prototype (also see section 3.5.4, Figure 60, p.96), users populate ZOIL's information landscape with small virtual notes to discuss them and to organize them into meaningful arrangements (*Geyer, Pfeil, Höchtl et al. 2011*). Geyer et al. designed the necessary interaction techniques with the goal to imitate collaborative

clustering actions at a real-world physical whiteboard as closely as possible through multi-touch manipulations. To facilitate the collaborative organization of notes into clusters, Geyer et al. included a simple clustering algorithm that automatically aligns and associates notes when released close to each other (see Figure 105, a-c). The resulting behavior can also be described using a metaphor of magnetism: Objects close to each other are attracted by an inter-object magnetic force that lets them snap together.

This metaphor can also be used to explain further interaction techniques in *AffinityTable*: To detach and drag a single note from a cluster, users can use a single finger to apply the entire dragging force on a single note to overcome the magnetic force between the note and the rest of the cluster. Dragging multiple notes of a cluster with multiple fingers at the same time, distributes the dragging force on the entire cluster. The dragging force stays below the inter-object magnetism and thus the entire cluster is moved instead of detaching a single note (see Figure 105, d-f).

As illustrated in the prototypes *Media Seminar Room*, *DeskPiles*, and *AffinityTable* there are many possibilities of providing users with space during sensemaking in ZOIL user interfaces. To better inform design choices, the user study in section 5.7 provides empirical data about how users adopted space in realistic usage situations and elicits their requirements for the way space should be provided and manipulated.

5.5 Marks and Annotations

Another important activity during knowledge work and sensemaking is placing marks and annotations on information items. Thereby space does not only serve as a medium for creating spatial configurations but also carries extra information in means other than formal visual syntax. Such a *"secondary notation"* (*Blackwell and Green 2003*) enables users to record things that have not been anticipated when designing a system and its primary notation. Rather than anticipating every possible user requirement, many systems support secondary notations that can be used however the user likes, for example to mark and highlight relevant pieces of information or to annotate documents or figures with handwritten notes or sketches.

Simple forms of a secondary notation are flexible uses of colors or format choices to indicate information additional to the content of text (*Blackwell and Green 2003*). Andrews et al. have observed the use of highlighting text for identification and extraction of passages in a document (*Andrews, Endert, and North 2010*). Highlighting, as opposed to creating separate text snippets, has the advantage of isolating passages without removing the information from context. The pattern of the highlights also provides a visual cue that aids recognition of items.

Kidd explores more advanced ways of storing and re-displaying arrangements of visible marks as a way of visually stimulating the recovery of an earlier mindset for a knowledge worker (*Kidd 1994*). Kidd recommends concentrating on capturing and reproducing the appearance of marks made by knowledge workers rather than interpreting them. *"These marks made on paper, screen (or indeed any other physical*

surface from cave wall to whiteboard) is how people change their environment in order to carry information from place to place or time to time (...). They are also used to externalise their own thinking – a type of scaffolding whilst they are in the process of informing themselves (...). Changing these marks or their arrangements may not do the knowledge worker a service when it comes to cueing the re-call of their current understanding of an issue or their intent to inform another" (*Kidd 1994*). In conclusion, Kidd calls for a re-evaluation of *"the ancient breakthrough of making written marks in the context of modern computing and communications"* (*Kidd 1994*).

For Tang et al., lightweight annotations also play an important role when working collaboratively on interactive tabletops (*Tang, Tory, Po et al. 2006*). Tabletop task spaces should support mobile, unobtrusive, and transient annotations, because annotations help to generate and track independent work. Tang et al. recommend the easy creation, mobility, and modification of annotations, since both spatially-relevant and spatially-invariant annotations were frequently used by the participants of their study.

The use of marks and annotations is particularly important in domains in which handwriting and annotation are established practices. For example, Oleksik et al. observed the work practice in the NanoPhotonics Centre of the University of Cambridge and found that all researchers used tablet PCs and OneNote electronic notebooks to which they added context through handwritten notes and sketches (*Oleksik, Milic-Frayling, and Jones 2012*). This great importance of pen input for marks and annotations later became a requirement for the ZOIL-based *DeskPiles* tabletop. It therefore supported precise marks and annotations with a pressure-sensitive stylus on tablet PCs and coarse-grained highlighting using fingers on the tabletop similar to a highlighter pen.

5.6 Examples for Providing Marks and Annotations in ZOIL

The ZOIL example prototypes explore different approaches for providing users with the possibility to use marks and annotations within the visual workspace. The following sections introduce different approaches to demonstrate the possible design space. Thereby the design space is structured into three kinds of marks and annotations: marking and highlighting, notes and sketches, and visual links.

5.6.1 Marking and Highlighting

To illustrate possible interaction techniques for marking and highlighting items, this section uses the ZOIL example prototype *DeskPiles*. As shown in Figure 81 (p.127), users can use semantic zooming to access the content of information items (e.g., a presentation slide) and a tool palette with different tools for marking and highlighting (Figure 106). After zooming out, the tool palette disappears while the marks and highlights remain visible to achieve the effects of visual cues that aid recognition of items described in (*Andrews, Endert, and North 2010*).

The interaction with the tool palette and its tools is similar to that of conventional paint or drawing applications. However, different device-specific input events are supported.

For example, the tools can be used with touch input on a Microsoft Surface or Windows 7 device, with pen input on a tablet PC, or with mouse or touchpad on a Windows desktop PC or laptop. This enables users to make use of the device-specific input capabilities, for example, they can use tablet PCs for fine-grained pressure sensitive input with a stylus. To enable marks and highlighting on all content items, the ZOIL framework provides a transparent control that can overlay arbitrary objects. Furthermore, this control ensures that ink strokes and highlights are persisted and synchronized in real-time, so that they can be used during scenarios of synchronous collaboration, e.g., during discussions or presentations like in section 2.2 (p.37), where multiple devices are used at the same time.

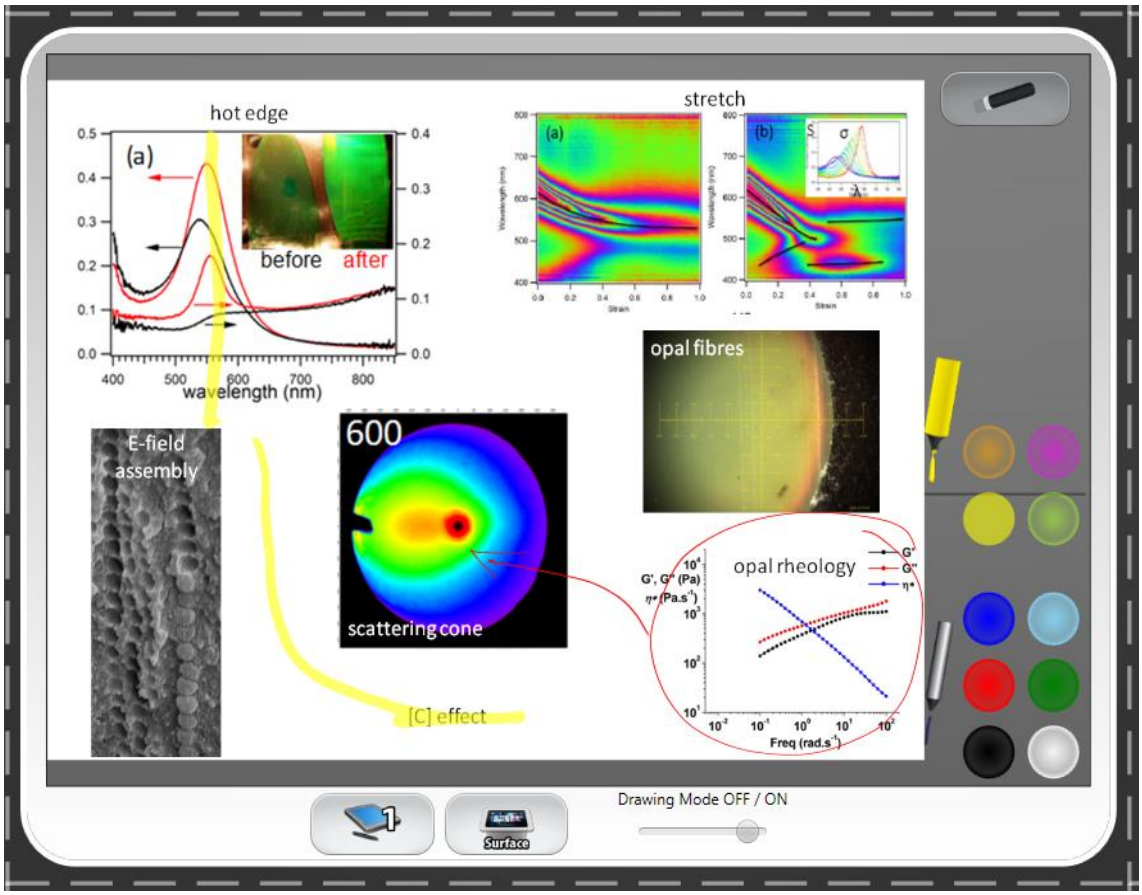


Figure 106 – *DeskPiles* provides a tool palette to annotate items with virtual ink and highlights.

5.6.2 Notes and Sketches

As described in sections 2.1 and 2.3, the *Media Seminar Room* and the *Distributed Sketching* prototype make use of Anoto's digital pen and paper technology to let users quickly create textual notes or sketches (see Figure 20, p.37 and Figure 30, p.44). Such virtual notes can serve as labels or as representations of new ideas and abstract concepts that are created during collaboration.

The *DeskPiles* prototype provides similar functionality for creating free-floating virtual Post-It notes. This can be done with every active device and at any time during the collaboration. By pushing an "Add Note" button on a tabletop, tablet PC, or desktop PC, a new note object is created above the grid. It can be used for writing or sketching with

touch input or stylus using the tool palette from Figure 106. If available, an on-screen or physical keyboard can be used to create notes in block letters. Since note objects in *DeskPiles* float above the grid, they can be freely positioned, resized, and rotated to express importance or relatedness to other objects in the grid. This functionality was extensively used during the user study in section 5.7, as is visible in Figure 111 (p.176).

5.6.3 Visual Links

DeskPiles also enables users to create visual links between objects as a further category of marks. Users can create links between any two objects in the workspace. They appear as curved lines and their main purpose is to visually express a semantic relation between two objects in those cases in which proximity cannot be used because of a great distance between the objects in the workspace. In addition to their role as visual cues, the links also enable an accelerated navigation within the workspace: By tapping on a link, an animated zooming and panning is issued that moves the link's destination object into the center of the screen. Similar to the note objects, visual links were used extensively during the user study in section 5.7, as is visible in Figure 111 (p.176).

5.7 User Study: DeskPiles

As a part of the *DeskPiles* project (see section 2.2 and Figure 27, p.42), we[57] conducted a combined formative usability evaluation and exploratory user study in the NanoPhotonics Centre of the University of Cambridge (henceforth NP). As described in the usage scenario in section 2.2, long standing work practices in NP required that researchers present overviews of recent work during group meetings, which users discuss and collectively interpret. The creation and use of these research overviews closely resembles the concept of a *knowledge crystallization task* by Card et al. (see section 1.5.4, p.17) including a phase of sensemaking. Apart from studying the use and reuse of information artefacts among the members of NP, we focused on the question whether the benefits of space for sensemaking and the importance of marks and annotations that were identified in previous work (*Andrews, Endert, and North 2010*; *Frank M. Shipman, Hsieh, Maloor et al. 2001*; *Kidd 1994*; *Kirsh 1995*) can also be observed when using a ZOIL-based prototype and what implications this has on the design space of ZOIL user interfaces. For this reason, we adapted the *DeskPiles* prototype to support Card's *knowledge crystallization operators* and gave users realistic tasks to observe their sensemaking, marks, and annotations under realistic conditions.

[57] Any use of "we" in section 5.7 refers to me (Hans-Christian Jetter) as well as to Toni Schmidt (Master's student, University of Konstanz), Gerard Oleksik (user researcher at Instrata Ltd.), and Natasa Milic-Frayling (Principal Researcher, Microsoft Research Cambridge). *DeskPiles* was originally conceived of and designed by me together with Master's student Toni Schmidt with whom I also implemented the prototype. Additional design and implementation was provided by Jens Gerken and Michael Zöllner.

For the study, I and Toni Schmidt adapted the prototype to the study design and tasks. The study design following Card's knowledge crystallization operators was the result of discussions with Gerard Oleksik and Natasa Milic-Frayling. Gerard Oleksik was the primary experimenter during the sessions and led the interviews. In some cases, I asked additional questions as secondary experimenter and provided help with the prototype. Gerard Oleksik analyzed the greatest part of the collected data and maps for studying the use and reuse of information among members of NP. I used and extended his analysis to extract design implications for the ZOIL paradigm.

5.7.1 Support for "Acquire Information"

In order to *acquire information* (see section 1.5.4, p.17), files containing relevant information items can be dragged from the Windows file space into *DeskPiles*' zoomable visual workspace that contains the grid (see Figure 22, p.38). After dragging and dropping a file into a grid cell, *DeskPiles* creates paginated visual proxies of the document content inside the destination grid cell (see Figure 104, p.165), so that content can be navigated by zooming and panning without the need to start an external viewing or editing application.

DeskPiles features native support for commonly used file types within the NP environment such as PPT for PowerPoint presentations, JPG for photos and microscopic imagery, or PNG for screenshots taken from lab software. Due to technological constraints (see section 3.5.7, p.102), *DeskPiles* does not natively support PDF documents or OneNote notebooks, which are also frequently used in the NP environment. As a workaround, users can print or save such content into Microsoft's XPS document format which *DeskPiles* natively supports. Other unsupported file types can be dragged directly into the grid where they then appear as an icon but not as paginated content. After zooming into these icons, users can push a button to open the source file from its original location with its associated application.

DeskPiles does not explicitly support Card's three sub-activities of acquiring information, i.e., *search*, *monitor*, or *capture*. However, through the interoperability between the tool and the Windows operating system such functionality can be provisionally integrated by dragging and dropping content from external application windows.

5.7.2 Support for "Make Sense of It"

The zoomable visual workspace supports users in three of the four sub-activities of *sensemaking* identified by Card (see section 1.5.4, p.17): It acts as a unifying visual metalayer above the file system and applications that enables users to *integrate different sources* regardless of the content's origin, location in file space, or file format. In the visual workspace, the information items can be clustered, sorted, and shifted around to *find a schema* and rearranged, visually linked, or annotated to *re-code information into schema* during an iterative collaborative process.

DeskPiles does not explicitly help users with the fourth sub-activity of *extracting information* yet. However, standard Windows features such as copy & paste, creating screenshots, or snipping tools enable users to create intermediate artefacts in file space that can serve as information extracts.

5.7.3 Support for "Create Something New"

The spatial arrangement, linking, and annotation functions that support sensemaking also enable the sub-activities to *organize for creation* and *author*. Ideally, the visual workspace develops over time from a collection of few items into a visual artefact that maps the relations and links between many information items from different sources

and projects. Furthermore, annotations can be used to integrate additional information such as open questions, conclusions, decisions, or action points.

5.7.4 Support for "Act on It"

The created visual artefact or map enables users to *distribute*, *apply*, and *act*. Indeed, distribution is possible by sending the map as a data file or publishing it in the Windows network using ZOIL's object database server. When opened inside *DeskPiles*, the map can then be used as a visual document on its own right or can serve as a kind of "launch pad" or as context-rich bookmarks for accessing the contained files and folders in Windows Explorer or associated applications. Distribution can also happen during meetings or briefings where the map is used as a spatial and zoomable presentation medium.

Since initial observations in the lab facilities revealed that projectors, large vertical screens, or electronic whiteboards are often available in close proximity, *DeskPiles* also enables access to maps from this variety of available devices. This facilitates sub-activities such as *apply* or *act* by providing conclusions and action plans from meetings in the context of where they should be put into practice.

5.7.5 Study Design

The study involved four participants working in NP. The number of individuals was limited due to the high demand on the users' schedules. We recruited a Post-doc working on multiple projects (participant P1), a third year PhD student (participant P2), a first year PhD student (participant P3), and the research leader of the group (participant P4). During the sessions, we observed typical usage patterns and how participants used space for sensemaking and annotation. We also elicited requirements from users, asking them to provide feedback regarding the performance of *DeskPiles* and the features that would help improve it.

We designed three tasks to recreate *knowledge crystallization* that P1-P4 normally perform. Specifically, we instructed the participants P1-P3 to individually create an overview of their area of research and identify critical research questions arising from their work (sessions 1-3). This was followed by a group session (session 4), in which we asked P1-P3 to collaboratively create a consolidated overview of their respective research areas and identify shared research questions. In a final session, P4 actively reviewed the consolidated overview and was asked to comment on it and to make necessary alterations and annotations (session 5).

All tasks were held *in situ* at the research leader's office at the field site where group meetings were regularly held before. Each of the 5 two hour sessions was broken into 90 minutes use of the tool and a 30 minutes semi-structured interview. All sessions were video-taped, and transcribed for analysis. Furthermore, we automatically took images of the tabletop's screen every 15 seconds to enable us to record how the content representations evolved over time. Interviews focused on the artifact that the users created, experiences of using the tool, and eliciting further requirements.

Session 1-3: Individual Sensemaking (one user per session).

As preparation for the session, participants P1-P3 were asked to individually gather a list of information items and resources representative for their individual work (*acquire information*). Once seated at the tool (Figure 107), each participant was asked to organize these items using the *DeskPiles* prototype (*make sense of it*) to create a spatial representation of their project (*create something new*) that they would present to their colleagues in session 4 (*act on it*).

Figure 107 – Session 1-3: single participant sits at Surface for individual sensemaking.

Session 4: Collaborative Sensemaking (one session with three users).

In sessions 4, P1-P3 were asked to sit around the tabletop to create a consolidated project map using their individual maps as reference. In order to provide participants with visual access to their individual project maps, we projected the project maps from the previous sessions onto the wall and thus provided a shared display (Figure 108). If participants wanted to switch the projection to a particular project map or zoom in onto a particular area, they would direct the request to the experimenter conducting the study. During the session, participants selected and imported information from their own maps (*Acquire information*), which they collaboratively organized (*Make sense of it*) into a consolidated representation (*Create something new*) that they presented to their research leader (*Act on it*).

Figure 108 – Session 4: setup for collaborative sensemaking with additional vertical projection of individual results from sessions 1-3 (left). Three participants sat around tabletop and used their laptops/tablet PCs and the projection to create a consolidated project representation (right).

Figure 109 – Session 5: Active review of the consolidated map by the research leader.

Session 5: Active Review (one session with four users).
P1-P4 sat around the table (Figure 109). The session began by asking the research leader (P4) to interpret the shared project map from session 4 and then a spokesperson from the group of P1-P3 explained any outstanding aspects of the map. The research leader was then invited to make additions to the map (repeating the knowledge crystallization operators identified in previous stages).

5.7.6 Results: Use of Space and Creation of Maps

Although *DeskPiles* is very different from the traditional use of physical or digital lab books, participants quickly found ways to depict concepts and relationships and highlight questions from their lab work. Once participants had imported information items and resources into the tool, they quickly adapted to the spatial paradigm. They referred to the task as *"creating a map"* and developed methods of *"reading"* the map and handling the contained data. The tool appeared to quickly become secondary once the process of map creation had begun, especially in sessions 2 and 3.

Across all sessions, participants imported a total of 34 information items into maps. The participants primarily used excerpts from existing files by selecting individual pages, slides, or images, for example, by using print screen and image cropping tools. There were practical considerations for the use of such document excerpts. Participant P2, for example, purposefully selected two of nine sample images from a scientific poster to remove distracting information and, by this, reduce the need for zooming & panning.

Participants exhibited one of two approaches when constructing individual maps: P1 first imported all the information items and then began to cluster them according to area. P2 and P3, in contrast, interspersed importing and clustering items, only importing further items after finishing clustering existing ones. The process of spatially laying out, arranging and rearranging items helped participants clarify their thinking about their work. Participant P3 referred to the activity of moving items as "*a level of mental organization almost like arranging something on your desk (…) you can create a bit of a flow*". Spatial groupings of items helped participants gain overviews of their work. Speaking of his completed project map, participant P1 stated that "*being able to see the three groups of work is really quite useful… it's like a memory tool almost, to remind yourself about what's going on. So, let's not forget how that connects with this piece of work over here*".

As expected from findings by (*Andrews, Endert, and North 2010*; *Frank M. Shipman, Hsieh, Maloor et al. 2001*), participants used space as the main organization principle to group information items. They created 'zones' in the grid space by either clustering items in specific areas or laying out items vertically to create columns. During the session they articulated the specific criteria they used to create the zones in the grid space.

Participant P1 used the grid to separate commercial and scientific work by creating a single column for commercial work on the left, and a cluster of items related to the scientific work on the right. Near the center he placed a sketch that he had prepared as a

high level overview (Figure 110). Within these zones, P1 used more complex spatial semantics. For example, within his commercial grouping he arranged items chronological from top to bottom (Figure 110, left) similar to the use of timelines that Andrews et al. observed in their study (*Andrews, Endert, and North 2010*).

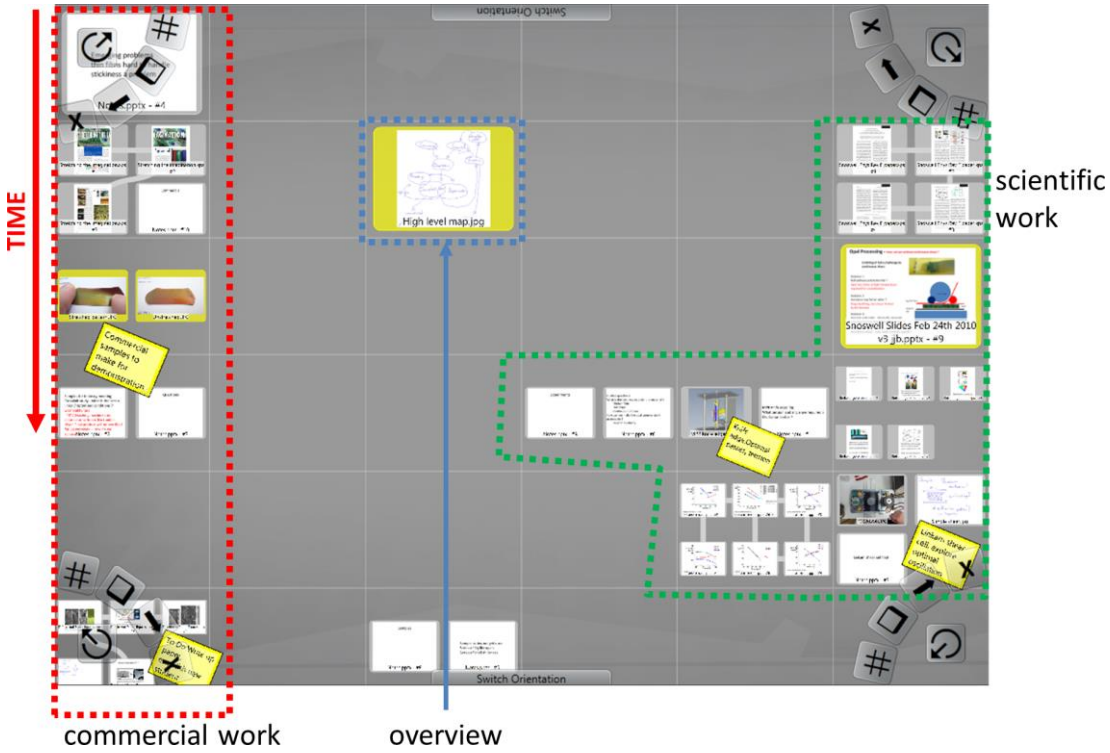


Figure 110 – Map created by P1 during session 1.

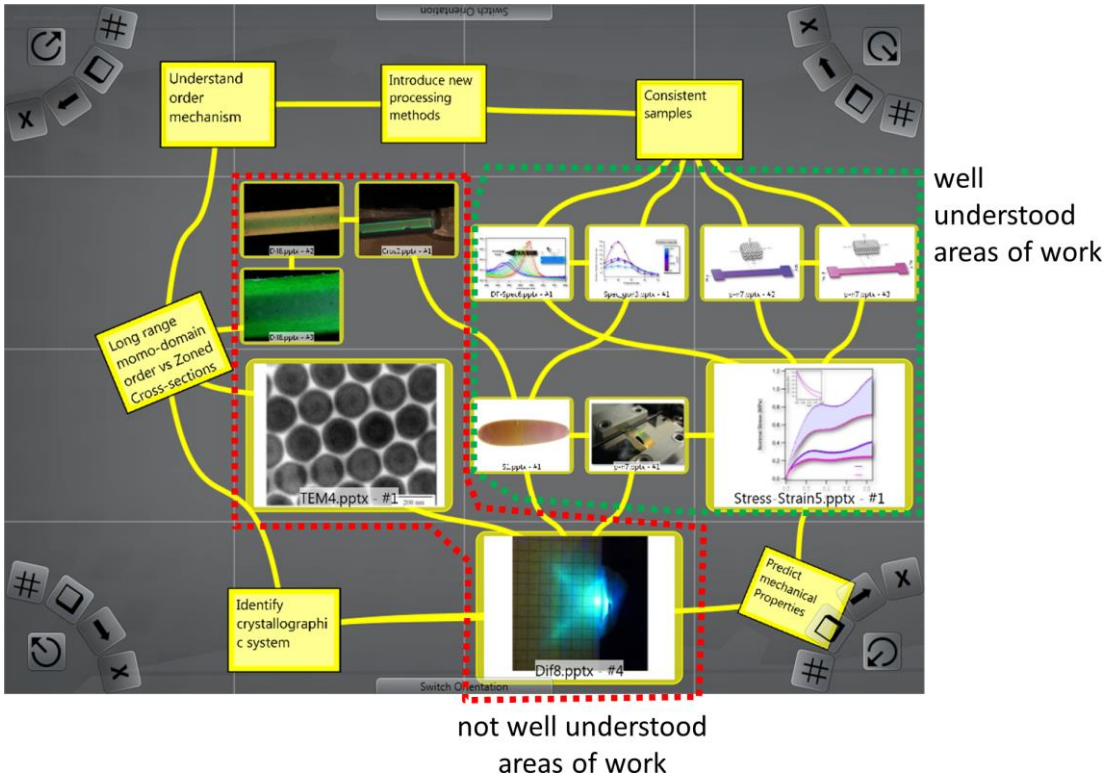


Figure 111 – Map created by P2 during session 2.

Participant P2 created three broad semantic zones that were not spatially demarcated (Figure 111). In the center to the right he placed information items related to well understood areas of work. Below and the left of this, he placed items related to areas of research which were not well understood. He used the surrounding cells to explain the relationship and dependencies between items in both zones and also to describe typical phases of lab work and cycles of knowledge generation in his project. P2 felt that spatial grouping had potential for identifying new links between areas of research. For example, this could be the case whenever items are arranged closely together in a map, but not so in other organization structures.

Participant P3 created two prominent semantic zones, corresponding to two different sides of his project ("scattering" and "emission"). Furthermore, a smaller area in the bottom left corner was used for future ideas (Figure 112). Like participant P1, participant P3 used spatial ordering from top to bottom within the "scattering" and "emission" zones. Each zone's content vertically lists the problem and possible solutions on the top, next research steps in the middle, illustrations of the experimental setup in the third row, and images of the collected experimental data on the bottom. Laying out his main work areas in adjacent columns helped participant P3 identify dependencies between the experimental set up in one area of his research and the questions and approaches driving the other. He stated that he would not have been able to do so within the linear form of a text document or PowerPoint presentation, which did not allow him to jump quickly from one set of ideas to another.

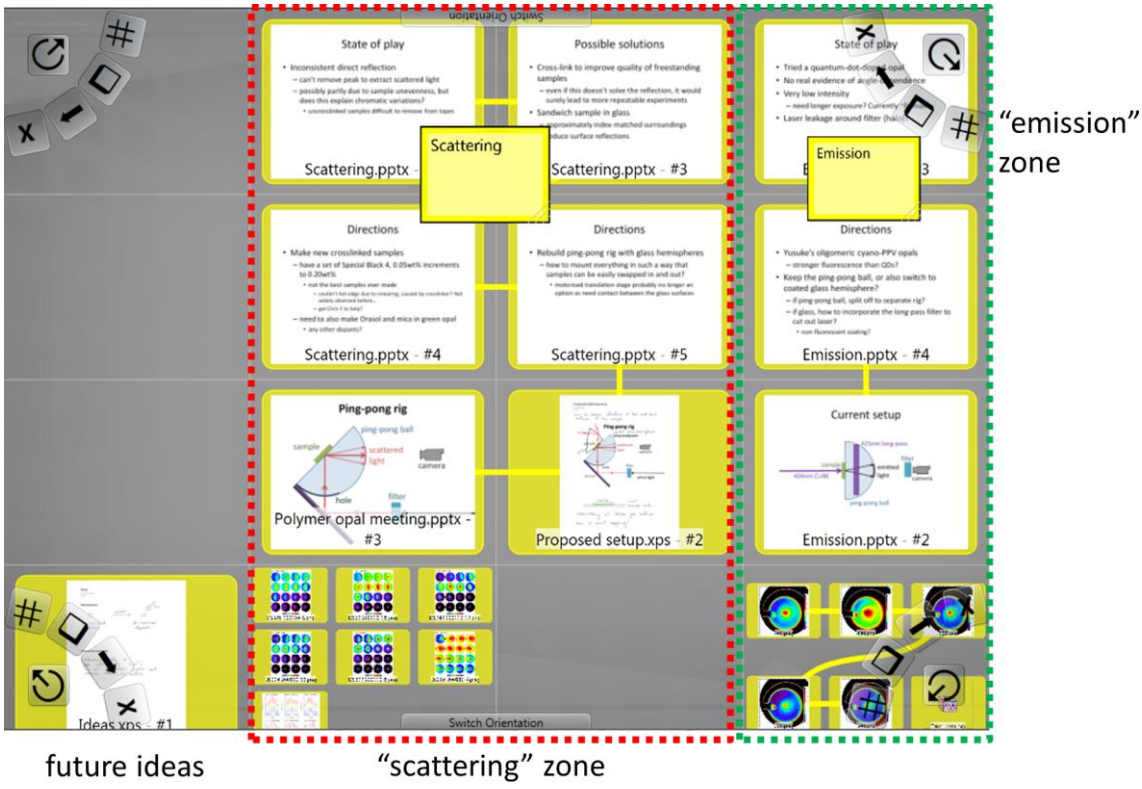


Figure 112 – Map created by P3 during session 3.

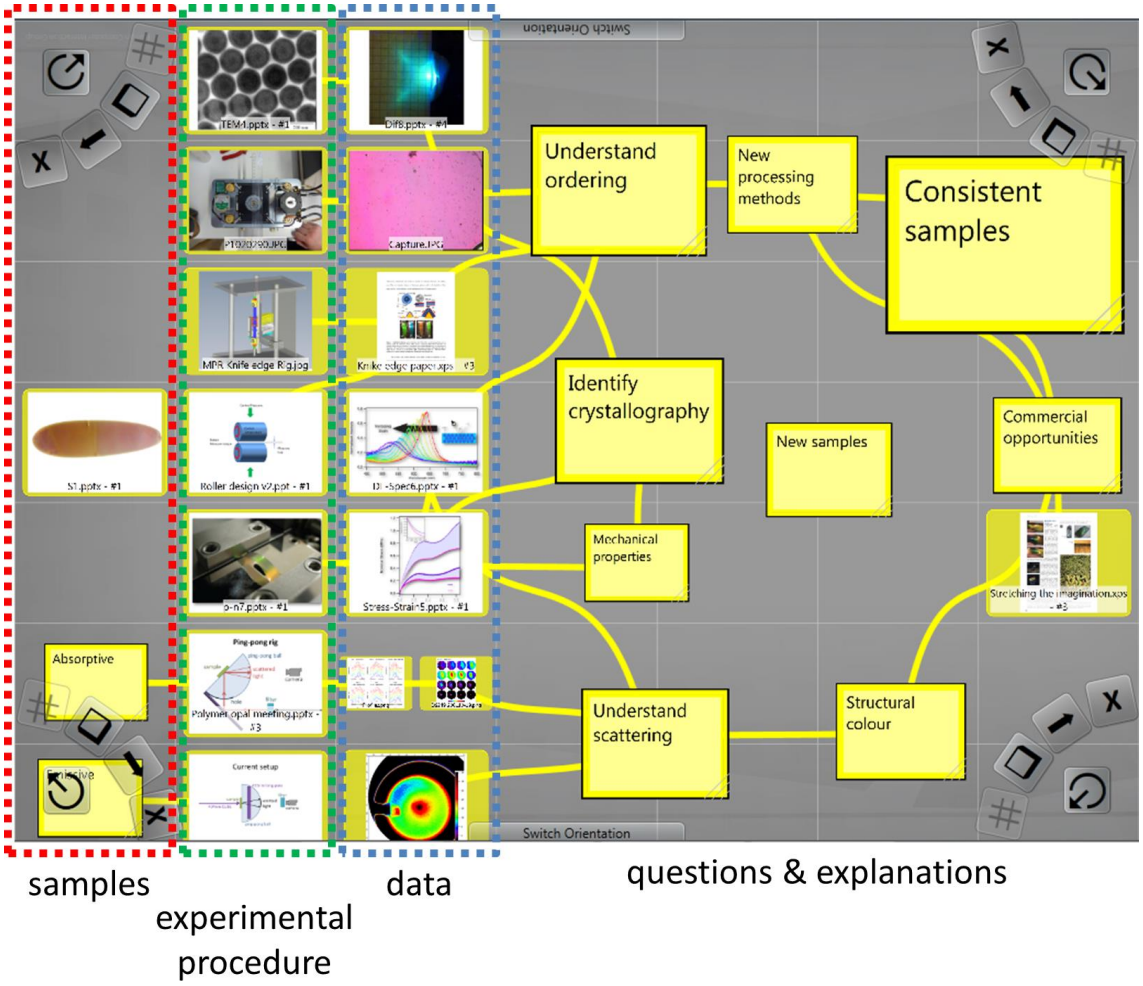


Figure 113 – The shared project map created by P1, P2, and P3 during session 4.

During session 4, participants P1, P2, and P3 created a shared project map (Figure 113) and quickly established a spatial organizational schema for arranging information items in the map. This schema, established by P1 after the first two items had been imported, consisted of the left to right ordering of 3 adjacent vertical columns representing material samples, experimental setup, and data, and a fourth space. The broad fourth zone designated a space for questions and explanations. Once the schema had been established, participants took turns importing and arranging their individual items accordingly, using individual project maps, projected on the wall, as a memory aid.

In this consolidated map, many of the items and resources from the individual maps from sessions 1-3 were reused. However, the amount of items per person was reduced and mostly visual items were selected, e.g., sketches, graphs, or photos. They were chosen as *pars pro toto* representations for expressing higher level abstract concepts and setting the ground for meaningful connections for two reasons: 1.) a single image could represent an entire class of items and 2.) an image could communicate information quicker than text or an entire document. Based on the participants' feedback, this use of images as *pars pro toto* representations appeared to bi-pass information processing required for reading text, thus lowering the threshold for comprehending connections

between concepts. For example, participants used a sample image from participant P1's lab notebook to represent all the samples used in experiments. Later in the session, participants discussed changing this image for something more aesthetically pleasing but decided against stating that *"we all know that shape and we all know what it is"*.

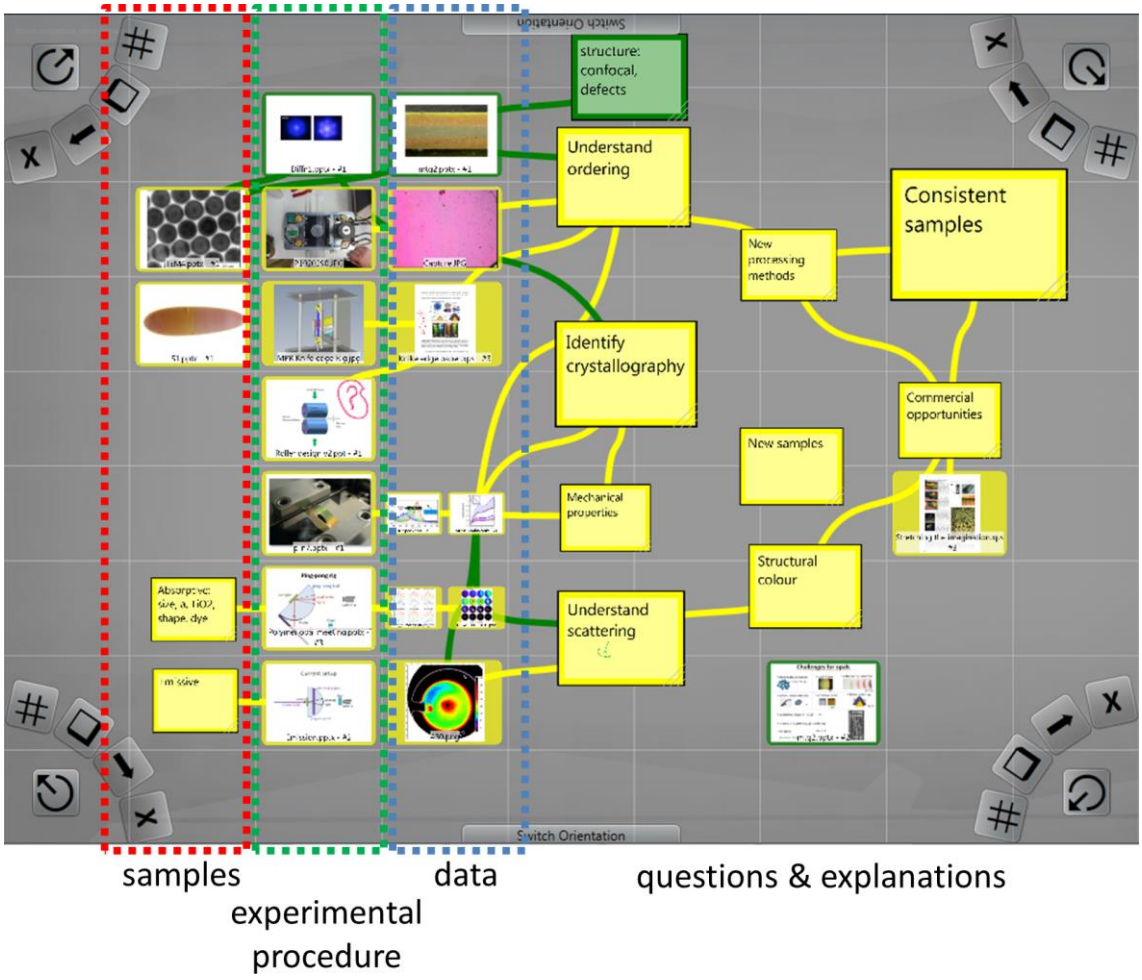


Figure 114 – The shared project map after active reviewing by the research leader P4 during session 5.

In session 5, participant P4 actively reviewed the consolidated map on the tabletop (Figure 114). During this session, the meaning of the visual objects and the layout was less fixed than in the previous sessions. While the visual objects had powerful symbolic value, their meaning sometimes was ambiguous: Participant P4 misinterpreted two of the icons. In the first case, he misunderstood that the specific sample should stand for all the samples and in the second he interpreted the icon broadly and the precise meaning only became clear during discussion. In the majority of cases, however, P4 correctly interpreted the meaning of icons and added further icons and notes to expand the scope of the map. To achieve this, he also increased the size of the map.

The role of visual icons as *pars pro toto* representations was most clearly articulated by P4 when asked how he understood some of the visual items in the consolidated map: *"it's definitely the concept (…) at this level I'm not thinking of the document underneath (…) so it's really concepts we're linking not data"*. P4 also reflected on the value of such icons:

"Even the big questions are hard to keep in my head and I instantly get it from the icons. You know that immediately… (…) because I see it this way, I'll think, oh, maybe we should do that, we should do that [indicating an icon at top of map] with that [indicating icon bottom of map] to have a look at that [indicating a third icon in the middle of the map] and I wouldn't get it from the words (…) it's not quick enough that you can make these links across the space - so that's the reason we might do it."

5.7.7 Results: Use of Annotations, Links, and Marks

In all cases, participants used *DeskPiles*' note objects to label or augment information items with textual annotations:

P1 positioned notes over information items to denote research intentions and questions related to them. He rotated the notes slightly clockwise to visually segregate them from the information items inside the grid and by this imitating the look of real-world tags (Figure 110). P2 positioned notes around the outside of the map and created links between them and the map's items. He used notes as explanatory labels to provide context and argumentation to the network of links he created (Figure 111). P3 used notes in front and between items to label clusters with the project they are related to (Figure 112). During the consolidation session, participants created and linked a series of notes to indicate questions arising from their projects (Figure 113). They arranged notes into two spatial categories to indicate progress already made within projects and questions they could ask building upon this. Participants also resized notes to indicate the priority of questions. Furthermore, they created two notes to represent special types of samples for which no images were available. These notes provided the framework for argumentation for session 5, during which it was extended with a further note by the research leader (Figure 114).

Based on these observations, notes are particularly important for quickly capturing abstract concepts, new ideas, or resources during collaboration that do not exist as objects in documents or the file system yet. Notes also help to reduce ambiguity and increase the readability and comprehensibility of a spatial map for others as they provide meta-level information about users' intentions during its creation. For example, they were used to label or explain relationships that do not become immediately obvious through spatial configuration alone.

In a similar manner, participants used visual links to express relations and semantic connections that could not be expressed by spatial proximity alone. Visual links designated complex interrelationships between objects over distance. For example, P2 made extensive use of linking and created 20 links to form a dense network of connected items and notes (Figure 111). Whereas other participants created relatively simple groupings, P2 created a complex representation, with interconnections across spatial groupings. Similar use of visual links is seen in the flow of questions in the consolidated map (Figure 113). Participants jointly created a large number of links to form connections between information items and the note items that contain research questions and goals. Since items were grouped in the left part of the map, links were

used as connectors to the explanatory notes because of their layout in the two-dimensional plane. In the review session, P4 further refined the map by creating additional links between existing items as well as new links between existing items and the ones he brought into the map (Figure 114).

While there were no differentiations among the types of links available in *DeskPiles*, we noted that they effectively connected and chained ideas and resources. However, participants expressed dissatisfaction with their relatively simple nature. Participants expressed desire for a richer linking language, including animated links to show flow of data, directional links, colored links, and links of different thickness to indicate critical and non-critical paths. During session 5, participants also suggested to assign links to different categories, so that the visibility of an entire category can be selectively turned off and on to reduce visual clutter when creating dense network of links.

The value of marks and highlights became most obvious during session 5. The research leader used simple markings to express which parts of the map he found hard to comprehend or ambiguous and that should be revised by the other participants. For example, he marked an unclear figure with a question mark (Figure 115).

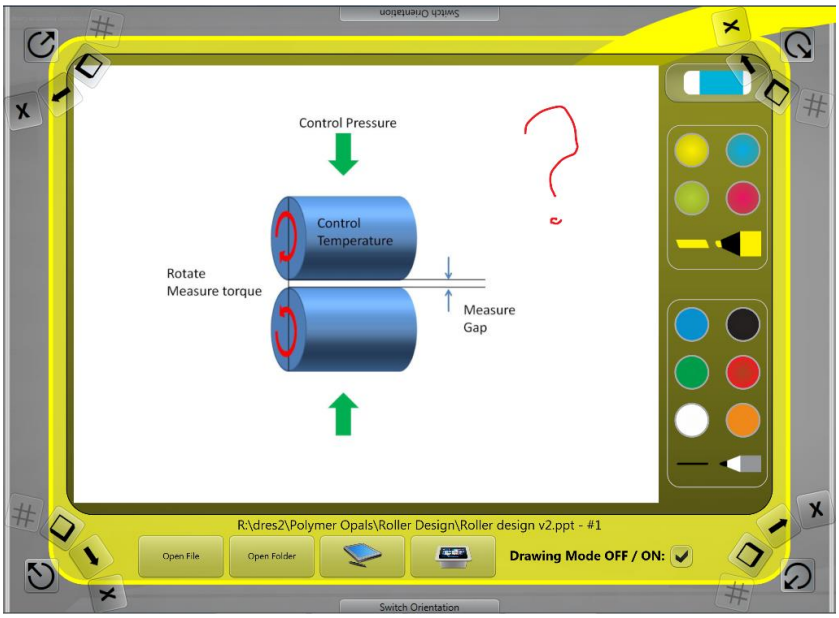


Figure 115 – P4 used the tool palette to add a question mark to an unclear figure.

During session 5, we also observed a more complex use of marks by P4. During the review session, an image with experimental data caught his attention (Figure 116). P4 zoomed into the details of the image and drew elliptical shapes around the centers of the different plots to highlight the orientation and size of structures in the data and discuss them with the other participants. During this discussion, he used further highlights to explain what orientation and size he would expect and how the images in the map differ from his expectations (Figure 117). In this case, the drawing and highlighting tools did not only serve to add meta-level information to the map but became tools for creating awareness during discussion and for a first rough visual analysis of data.

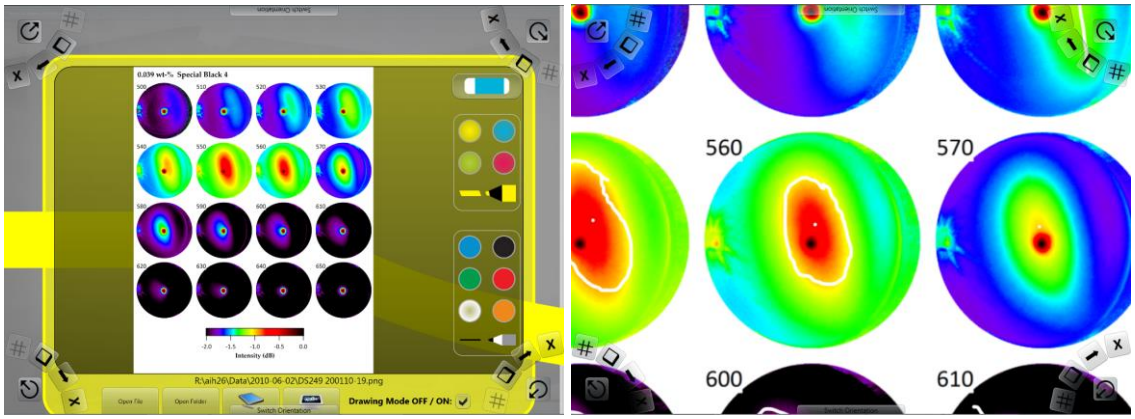


Figure 116 – The image of experimental data that caught P4's attention (left). After zooming in, P4 added white elliptical shapes to the image to highlight interesting visual structures (right).

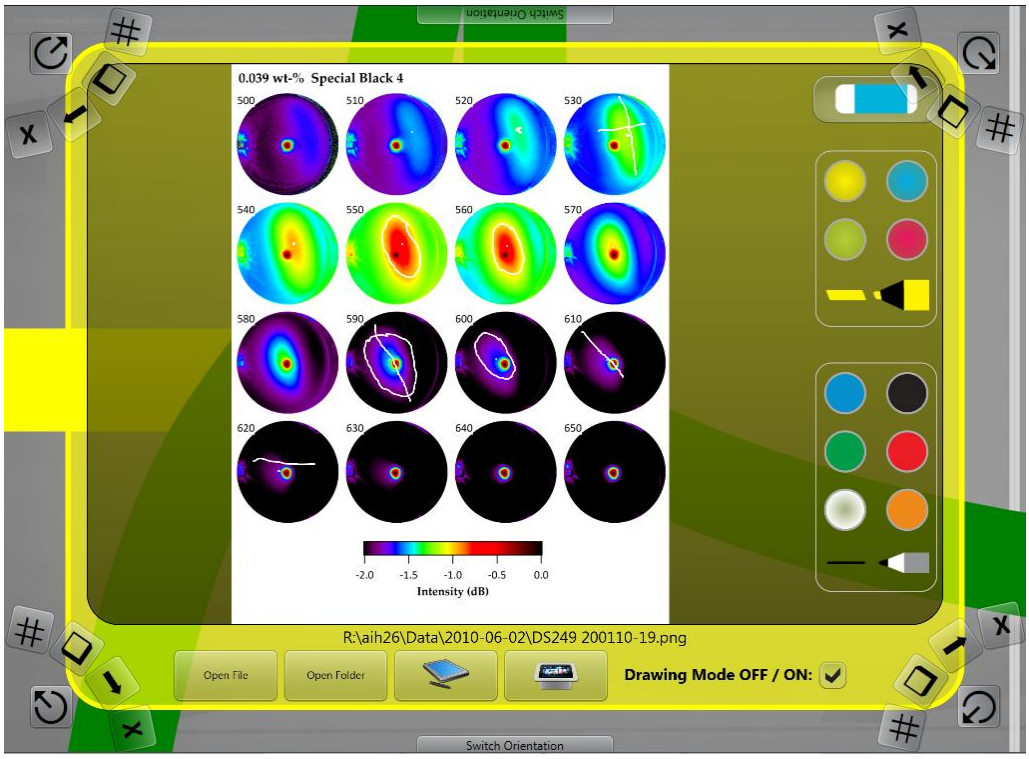


Figure 117 – During discussion, P4 added a series of ellipses and straight lines to the image to highlight the difference between expected and measured orientations in the data.

5.7.8 Miscellaneous Findings

Participants' feelings towards *DeskPiles*' multi-scale snap-to-grid mechanism were mixed. P2 felt that it helped him arrange resources when *"not exactly sure how they're going to be organized"*. However, once he had established resource categories, he no longer felt the grid necessary. All other participants found the grid constraining and wished to arrange items in a more *"organic"* and *"freeform"* space, e.g., by drawing lines around resource clusters to create categories and overlapping categories. Furthermore, although the grid could be enlarged by further cells at any time, the participants often stopped adding content when they felt that adding items would overload the default size of the grid. This again resonates with the participants' desire for a more freeform space.

While participants felt that the ability to open source documents and locations from items in the map was beneficial, it became clear that they wished to access a far wider body of information. Participants expressed the desire to *"drill down"* into information such as experimental data, experimental conditions, related lab notebook pages and related papers. P1 suggested that zooming into a cell could give the option to link to a wide variety of related information sources. Furthermore, participants expressed desire to nest maps or create links between them, as a method of connecting and navigating between related work spaces. P4 also wished to browse through the chronology of meetings maps and related data with a timeline or 3D navigation paradigm using overlaying maps to navigate in time or the process of idea development.

5.7.9 Implications for Design

Based on our findings, I extracted following implications for the design of ZOIL user interfaces and ZOIL design principles:

The process of spatially laying out, arranging and rearranging items can helps users to clarify their thinking about their work. Spatial groupings of information items help to gain overviews and to identify new links between different areas of work. ZOIL UIs should support this process by enabling the arrangement of items in an "organic" and "freeform" space and creating spatial semantics such as (chrono-)logical orders within clusters. This demands a considered choice of layouts and interaction techniques that are good tradeoffs between entirely free arrangements like in scatter views and more restrictive structures like grids. Furthermore, ZOIL UIs should enable users to quickly jump from one set of ideas to another and avoid linear forms of information presentation and navigation like in text documents or slide decks. They should also allow users to create links between different landscapes or to nest them, so that zooming enables a visual drill down into a wide range of related information.

Since some relationships cannot be expressed using spatial proximity alone, visual links should enable users to express complex interrelationships between objects over distance. ZOIL UIs should provide a rich linking language, including directional links, colored links, and links of different thickness to indicate critical and non-critical paths. Furthermore, introducing different categories of links whose visibility can be turned on and off could help to reduce visual clutter when the network of links becomes dense.

ZOIL UIs should provide functionality to create and integrate notes. First, notes are important for representing new abstract concepts, ideas, or resources that do not exist as objects yet and have to be generated during collaboration. Second, notes are important for labeling or providing context and other meta-level information to make the resulting information landscape more comprehensible to others. The interaction design must take into account both roles, because in the first case notes should imitate the look and behavior of regular items, while in the second case, their visual appearance and interactive behavior should clearly reflect their role as meta-level information items.

ZOIL UIs should support different user goals when importing items. When importing items, users do not always intend to include entire documents, media objects, or other large packages of information. In many cases, they want to use only smaller parts (e.g., images, figures) as self-sufficient pieces of information or as *pars pro toto* representations for an entire class of objects. When importing items, ZOIL user interfaces should offer the rapid selection and extraction of such smaller parts.

Providing tools for drawing, marking, and highlighting in a ZOIL UI is essential. They do not only serve to add meta-level information but also to actively point out the visual structures to discuss. Furthermore, they should enable a first rough visual analysis by letting users draw into diagrams and figures to highlight interesting parts and how they differ from their expectations.

5.8 P#3: "Provide Space for Sensemaking, Marks, and Annotations."

As a summary of the first part of this chapter, I formulate the *3rd ZOIL design principle* as follows:

"Provide space for sensemaking, marks, and annotations."

This is based on following summary of the discussion, observation, and findings in the first part of this chapter:

1.) Based on (*Kirsh 1995*), space is an integral part of human cognition and a key resource for collaborative knowledge work. Space is more than just a navigational cue or an extension of memory. It reduces time and memory demands by spatially decomposing complex tasks. Space also helps users to make sense of inputs. In our study, the process of spatially laying out, arranging, and rearranging items helped participants to clarify their thinking. Spatial groupings of items helped to gain overviews that would not have been able to achieve using non-spatial linear forms like text documents or slideshows. Therefore, ZOIL UIs should provide users with the freedom to flexibly arrange, cluster, or configure content and functionality in space and scale instead of relying on static and restrictive spatial layouts.

2.) Space can serve as a temporary holding pattern for potentially interesting inputs and ideas which cannot be categorized and for which there is no decision how to use them yet (*Kidd 1994*). In a ZOIL UI, users must not be forced to file or classify immediately, but should be enabled to establish zones and clusters as rough spatial categorizations for experimentation and reflection during a *"pre-linguistic stage of reasoning"* (*Kidd 1994*).

3.) Space enables experimental exploration without imposing a strict interpretation. The meaning of spatial relationships can evolve with understanding, so that at the start of the sensemaking process, there is no need for a premature choice of an inappropriate and restrictive organizational structure. Instead, ZOIL UIs should actively support the transition from freeform and loosely structured arrangements to more formal structures in the sense of *"incremental formalisms"* (*Frank M. Shipman, Hsieh, Maloor et al. 2001*).

This demands a considered choice of layouts and interaction techniques that are good tradeoffs between entirely free arrangements and restrictive structures.

4.) Users do not always intend to include entire documents, media objects, or other large packages of information. In many cases, they want to use only smaller parts (e.g., images, figures) as self-sufficient pieces of information or as *pars pro toto* representations for an entire class of objects. When importing items, ZOIL user interfaces should offer the rapid selection and extraction of such smaller parts.

5.) Space is not only a medium for establishing spatial relationships but it also carries extra information in a *"secondary notation"* (*Blackwell and Green 2003*), for example, annotations, marks, highlights, or visual links. The appearance of such meta-level information (position, colors, format choices) should be captured and reproduced by the user interface to provide context, cue the re-call of an earlier mindset, and aid recognition of items (*Andrews, Endert, and North 2010*; *Kidd 1994*).

6.) A ZOIL UI should provide tools for marking and highlighting. They do not only serve to add meta-level information and context (e.g., marking unclear or particular important content) but also enable simple collaborative visual analysis of data by drawing into diagrams and figures to highlight differences between expectations and data.

7.) A ZOIL UI should enable users to quickly create note objects as annotations. They are important for representing abstract concepts, new ideas, or resources that do not exist as objects yet and have to be quickly generated during collaboration. They are also important for labeling or providing context and other meta-level information to make the resulting information landscape more comprehensible to others.

8.) A ZOIL UI should provide a rich linking language, including directional links, colored links, and links of different thickness to indicate critical and non-critical paths. Furthermore, introducing different categories of links whose visibility can be turned on and off could help to reduce visual clutter when the network of links becomes dense.

5.9 Space for Collaboration

The following second part of this chapter discusses the role of space for collaboration. This role is introduced by first looking at how users partition physical space into territories to coordinate their collaboration at traditional tables. Then these concepts of territoriality are applied to the domain of non-physical collaborative spaces, specifically to ZOIL's information landscape. The result of this is ZOIL's camera concept or "device as a camera" metaphor that is illustrated in terms of its potential for enabling and coordinating mixed-focus collaboration.

5.9.1 Territoriality in Collaborative Tabletop Workspaces

Scott et al. studied users' spatial partitioning of their interactions during collaborating at traditional tables (*S. D. Scott, Carpendale, and Inkpen 2004*). They observed that people naturally partition their interactions on a table with little to no verbal negotiation into three kinds of territories: *personal territories*, *group territories*, and *storage territories*. They describe their spatial properties and functionality as follows:

Personal Territories

Areas directly in front of people are typically used as their *personal territories*. People restrict their personal territories to a "socially appropriate" area, generally refraining from using the table space directly in front of others. Personal territories allow people to reserve a particular table area, as well as task resources for their own use. They also provide a space for people to disengage from the group activity and as a "safe" place to explore alternate ideas before introducing these ideas to the group.

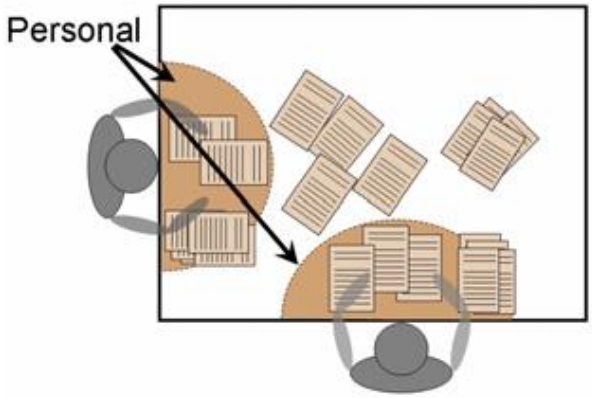


Figure 118 – Conceptual diagram for *personal territories*. Source: (*S. Scott 2005: 110*).

Group territories

Group territories typically cover any tabletop workspace that is not occupied by the personal territories. A group territory provides a space to perform main task activities and is also used to transfer task resources by handing off items via the workspace or depositing items for a partner to pick up later. Interactions inside the group territory can be described in terms of *tightly coupled* vs. *loosely coupled* interactions depending on the current coupling style of collaboration. This observation is also the origin of Tang et al.'s *mixed-focus collaboration* (*Tang, Tory, Po et al. 2006*) (see also section 1.4.3, p.13).

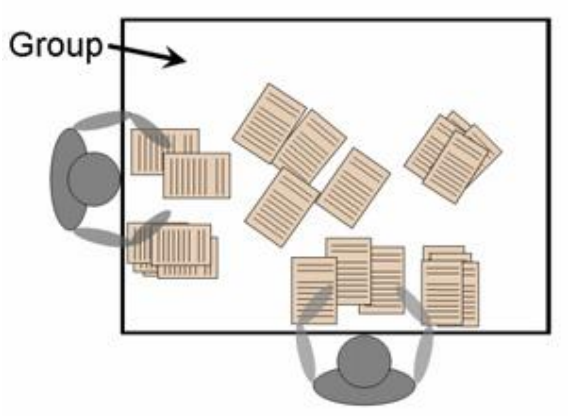


Figure 119 – Conceptual diagram for *group territories*. Source: (*S. Scott 2005: 110*).

Storage Territories

Storage territories are placed at various locations around the workspace, but generally migrate to the table edge as tasks progress. They serve as areas to store and to organize task resources not currently in use, so that persons can easily obtain them later when and where they need them. Typically, items in storage territories are loosely organized, but partial orders are maintained.

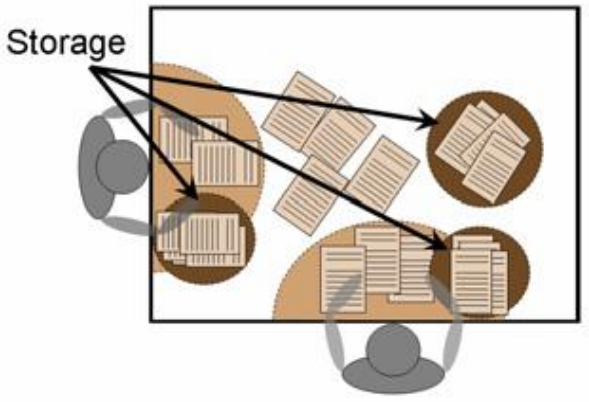


Figure 120 – Conceptual diagram for *storage territories*. Source: (*S. Scott 2005: 110*).

Space and Collaborative Coupling Styles

Scott et al. concluded that there is a close relation between the availability of space and collaborative coupling styles at traditional tables. This close relation could also be observed for interactive tabletop computers during the user study of *Facet-Streams* (see Figure 34, p.46) that is discussed in great detail in section 6.3.4 in the next chapter. During this user study, I made three observations:

In phases of tight coupling of actions, e.g., formulation or refinement of a single collective query by all participants, users moved the tangible facet tokens closer to the center of the tabletop and established a shared group territory there that was easily accessible for everyone from all sides.

When users' actions were only loosely coupled, e.g., during individually exploring the catalog content or revising personal criteria before adding them to the group's criteria,

the facet tokens were generally moved out of the center and more towards their current "owners" standing at the tabletop's edges. Sometimes they were even removed from the group territory and dragged into someone's personal territory to gain exclusive control.

The use of storage territories could be observed during phases of collectively reviewing query results. The tangible tokens were not used anymore during this phase and occluded the important result presentation underneath them. As a consequence, users moved them out of their field of view in the center towards storage territories along the edges of the tabletop.

As I discuss in section 6.3.4, in the case of *Facet-Streams*, the use of tangible user interface elements greatly facilitated the frequent spatial reconfiguration of tokens and thus also enabled fluid transitions between different coupling styles, e.g., between the different phases that are mentioned above.

Due to the close relation between coordinating collaboration and the use of space, Scott et al. concluded that tabletops should always provide appropriate table space. An inappropriately sized table may negatively impact the collaboration because collaborators may not have enough space to effectively disengage from the group activity or collaborators may need more explicit coordination to divide up an activity (*S. D. Scott, Carpendale, and Inkpen 2004*). Subsequent work by Tang et al. identified spatial overlaps of individuals' desired working areas as a common source of interference during tabletop collaboration (*Tang, Tory, Po et al. 2006*). For this reason, they suggest providing additional mobile high resolution personal territories, for example by using distinct displays for personal work such as mobile tablet PCs.

As illustrated by the *DeskPiles* and *Distributed Sketching* prototypes (see section 2.2, p.37 and section 2.3, p.43), ZOIL interactive spaces provide such additional displays, either on mobile tablet PCs or on multiple tabletops or vertical displays. This raises the question, how ZOIL conceptually integrates this additional space into a collaborative workspace and how this resonates with the natural territorial behavior of users described by Scott et al. and Tang et al.'s demand for a *"flexible variety of coupling styles"* and *"fluid transitions"* between them (*Tang, Tory, Po et al. 2006*).

5.9.2 ZOIL's Camera Concept for Collaboration

Unlike today's operating systems, ZOIL does not replicate the same screen content on multiple screens or stitches together the available physical displays to create a single logical screen with a greater, but still limited, number of pixels. Instead, in a ZOIL interactive space, each physical display serves as an independent camera to access and navigate ZOIL's information landscape, a shared zoomable workspace of infinite size and resolution. Users use explicit interactions such as multi-touch gestures or pen input for zooming and panning to control the display's content and to view and access the desired region of the information landscape at the desired scale (Figure 121, p.189, see also section 2.1, p.30). Since ZOIL uses semantic zooming (see section 4.3.4, p.125), it also

naturally adapts to the different screen sizes and resolutions (e.g., tabletops, mobile devices).

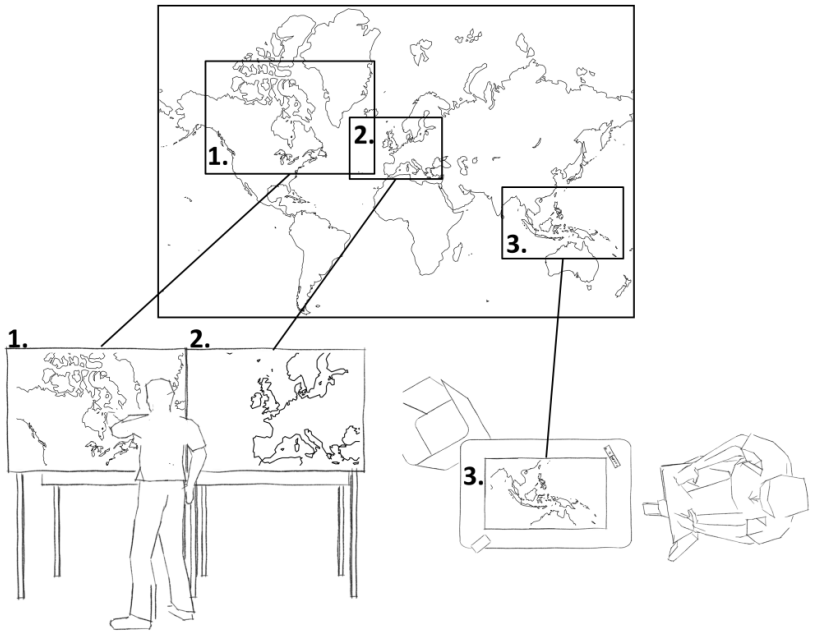


Figure 121 – ZOIL's camera concept or "device as a camera" metaphor: Each device 1-3 (bottom) shows an arbitrary region of the shared visual workspace (top) at an arbitrary scale and can be individually controlled using zooming and panning.

Unlike the majority of multi-display environments in literature[58], ZOIL does not necessarily use physical space and spatial relationships between screens as the *"referential domain"* (*Nacenta, Gutwin, Aliakseyeu et al. 2009*) to create a *"spatially continuous workspace"* (*Rekimoto and Saitoh 1999*) for cross-display interactions. Enabling such interactions, e.g., dragging objects between two screens with pointing devices or gestures, is not a part of the ZOIL design principles. However, such interactions have already been realized by others using the ZOIL software framework (*Fäh 2011*; *Jenabi 2011*). They demonstrate how a spatially continuous cross-display pointing can co-exist with ZOIL's camera concept. In future, this could facilitate the exchange of objects between distant places in the information landscape, particularly compared to portal or split screen techniques (see section 4.3.6, p.129). ZOIL also does not use spatially-aware mobile displays for *"peephole"* navigation in the physical environment (*Yee 2003*) or for creating rooms of mixed or augmented reality (*Butz and Krüger 2006*). However, a similar approach for navigating ZOIL's information landscape by physically moving a tablet in front of a large screen is currently explored (*Rädle, Butscher, Huber et al. 2012*), but is also not part of this thesis. Instead, in the context of this thesis, ZOIL's "device as a camera" metaphor is important, because it can support many different collaboration and coupling styles by enabling a flexible and natural *ad hoc* partitioning of an infinite virtual workspace:

[58] See section 3.4.7 (p. 73) for a discussion of related work about spatial paradigms for information access and different multi-display interaction techniques and software frameworks.

During loosely-coupled collaboration, users can use a personal device to zoom and pan to an empty area in ZOIL's landscape to establish a virtual personal territory there. In this territory, users have space to collect task resources for their own use, disengage from the group activity, and develop own ideas or intermediate results in a "safe" place. This becomes particularly useful when accessing personal territories with devices such as tablet PCs that provide device-specific output and input, for example with a pressure-sensitive stylus. By this, ZOIL satisfies Tang et al.'s 3rd design implication: *"Provide mobile high resolution personal territories"* (*Tang, Tory, Po et al. 2006*).

Whenever needed, users can move their view and their items between different personal, group, and storage territories at different locations in the landscape. Users can naturally establish territories in the virtual space of the information landscape simply by moving to a certain location in space and scale and starting to collaborate there. There is no need for explicitly creating new work zones or workspaces or setting up access rights. This enables users to quickly establish *ad hoc* group territories for tight collaboration, while other group members are currently working on their own in their personal territory or the rest of the group is working in a different group territory. Thereby, unlike on a physical table, the size of the workspace is infinite and enables users to create an infinite number of intermediate group or storage territories. This supports Tang et al.'s 1st design implication: *"Support a flexible variety of coupling styles"* (*Tang, Tory, Po et al. 2006*).

All the different territories coexist in ZOIL's information landscape without being separated by display or device boundaries. The information landscape provides a spatial frame of reference and spatial cues for navigating between them. At any time, users can move between the different territories and thus ZOIL provides the *"fluid transitions between coupling styles"* (*Tang, Tory, Po et al. 2006*) of Tang et al.'s 2nd design implication.

As discussed in section 1.4.1 (p.11), an important factor for successful collaboration is creating the necessary awareness. ZOIL's camera concept enables users to establish a shared display (e.g., a large vertical screen) that is visible for all collaborators and continuously shows a certain region of the information landscape for the purpose of increasing awareness. For example, in Figure 29 (p.43), a large vertical screen is used to zoom into a virtual Post-It note from the information landscape. This note is used by the collaborators for note taking, documenting output, and indicating the progress and state of the collaborative process. Such a screen could also serve to continuously provide a spatial overview of the information landscape and indicate which regions in it are currently accessed by the different collaborators with their different devices (see Figure 86, p.131). Therefore, ZOIL's camera concept can also help to increase awareness by letting users view *"visually persistent"* content from the information landscape on *"fixed function"* displays that play a similar role like the physical whiteboards with "task lists" or "bug lists" in traditional office or lab environments (*Tang and Fels 2008*).

5.9.3 Physical Tables vs. Zoomable User Interfaces

I believe that collaboration at traditional physical tables should serve designers of multi-device interactive spaces as a benchmark in terms of fluidity and naturalness. Observing interactions at physical tables can also inspire interaction designers how to better exploit users' pre-existing real-world skills, i.e. the *"environment awareness and skills"* and *"social awareness and skills"* of reality-based interaction (*Jacob, Girouard, Hirshfield et al. 2008*). ZOIL's camera concept is such an attempt to blend the familiar and natural partitioning of space when collaborating at traditional tables with the power of a virtual zoomable workspace of infinite size and resolution that can be accessed from multiple devices. However, at this stage of research on ZOIL, the assumed benefits of ZOIL's camera concept for coordinating collaboration are only theoretical and are not based on empirical user studies. They are derived from the findings from tabletop and multi-display literature such as (*Jetter, Gerken, Zöllner et al. 2011*; *S. D. Scott, Carpendale, and Inkpen 2004*; *Tang, Tory, Po et al. 2006*; *Tang and Fels 2008*) and are based on the hypothesis that collaboration can benefit from virtual space in the same or at least in a similar way in which it benefits from physical space.

As discussed in section 5.3 (p.163) this should be subject to further research. Andrews et al. suggest user studies to learn about the transferability of the benefits of physical space on large displays to the virtual space in ZUIs (*Andrews, Endert, and North 2010*). In particular, the increased cognitive load for view management in a ZUI could reduce the benefit of virtual space compared to that of physical space. Future user studies should therefore examine, if the virtual navigation in a ZUI has enough similarities with the physical navigation in front of a large display or large tabletop to recreate a similarly fluid and natural partitioning of the virtual workspace during collaboration.

5.10 P#4: "Provide Space for Coordinating Mixed-Focus Collaboration."

As a summary of the second part of this chapter, I formulate the *4th ZOIL design principle* as follows:

"Provide space for coordinating mixed-focus collaboration."

This is based on following summary of the discussion and design suggestions in the second part of this chapter:

1.) There is a close relation between the availability of space and collaborative coupling styles at traditional tables or interactive tabletops. People naturally partition their interactions with little to no verbal negotiation into spatial territories. Tabletops should always provide appropriate table space to avoid negative impacts on collaboration or interference due to overlapping territories (*S. D. Scott, Carpendale, and Inkpen 2004*; *Tang, Tory, Po et al. 2006*).

2.) To avoid interference in a ZOIL interactive space, ZOIL's collaborative workspace is a zoomable information landscape of infinite size and resolution. All active devices with a display (e.g., mobile devices, tabletops) can serve as individual "cameras" into this

information landscape that provide personal or shared views. Cameras can be individually controlled by the collaborators to zoom and pan to a desired region of the shared information landscape to view it at the desired scale. Collaborators can use this “device as a camera” metaphor of ZOIL to access different, same, or overlapping regions of the shared workspace at any time.

3.) ZOIL’s camera concept (or “device as a camera” metaphor) supports many different collaboration and coupling styles by enabling a flexible and natural *ad hoc* partitioning of the virtual workspace. Users can create an infinite number of personal, group, or storage territories in the landscape by moving to a certain location in space and scale and starting to collaborate there. The available virtual space and the spatial navigation between territories support many different coupling styles and fluid transitions between them during *“mixed-focus collaboration”* (*Tang, Tory, Po et al. 2006*).

4.) At this stage of research on ZOIL, the assumed benefits of ZOIL’s camera concept for coordinating collaboration are only theoretical and are not based on empirical user studies. Future user studies should therefore examine, if the virtual navigation in a ZUI has enough similarities with the physical navigation in front of a large display or tabletop to recreate a similarly fluid and natural partitioning of the virtual workspace during collaboration.

6 Post-WIMP Information Visualization

This chapter briefly introduces the fundamentals of information visualization (or InfoVis) and how InfoVis techniques can help users to manage today's growing number of functions and information items. A summary of relevant cognitive models from (*Elmqvist, Vande Moere, Jetter et al. 2011*) explains how post-WIMP interaction styles using multi-touch and tangible input can achieve an enhanced user experience of InfoVis with a *fluid interaction*. This is followed by an in-depth description of the design and user studies of *Facet-Streams* (*Jetter, Gerken, Zöllner et al. 2011*) which serves as a *"best-in-class"* example of post-WIMP and fluid information visualization. This chapter concludes with formulating the *5th and 6th ZOIL design principle* by extrapolating from the mentioned cognitive models and the empirical findings from *Facet-Streams*.

This chapter uses images of the ZOIL-based prototypes *EuroITV* (*Jetter, Engl, Schubert et al. 2008*) and the *HotelBrowser* prototype which is a result of the API usability evaluation presented in (*Jetter, Zöllner, Gerken et al. 2012*). Furthermore, it contains the cognitive account of fluid interaction from (*Elmqvist, Vande Moere, Jetter et al. 2011*) and includes the publication of Facet-Streams that received an honorable mention award at CHI 2011 (*Jetter, Gerken, Zöllner et al. 2011*).

6.1 Information Visualization

Information visualization or *InfoVis* uses computers to create interactive visualizations of abstract data such as numerical values, geolocations, networks, or hierarchies. Its goal is to give abstract data a visible spatial representation and to map data to visual variables such as position, size, shape, color, or orientation in order to make it perceptible for the users and to give them insight. Accordingly, *"Information visualization can be defined as the use of computer-supported, interactive, visual representations of abstract data to amplify cognition"* (*Card, Mackinlay, and Shneiderman 1999*).

This amplification of cognition is based on the users' visual perception and its *"broad bandwidth pathway into the mind to allow users to see, explore, and understand large amounts of information at once"* (*Thomas and Cook 2005: 30*). Because of this human skill, *"abstract information visualization has the power to reveal patterns, clusters, gaps, or outliers in statistical data, stock-market trades, computer directories, or document collections"* (*Shneiderman 1996*). To help users reveal such patterns, clusters, etc., it is important that InfoVis is interactive and that the underlying data transformations, visual mappings, and view transformations are not static, but can be dynamically changed by the users to satisfy their information needs (*Card 2008*). *"The purpose of visualization is insight, not pictures"* (*Card, Mackinlay, and Shneiderman 1999*).

One of the reasons that information visualization is becoming more and more important and receives more general adoption and integration into commercial products (*Plaisant 2004*) is the growth of the amount of digital information and tools that surround us. For Moran and Zhai the rapidly growing number of functions, applications, and files (hundreds if not thousands), puts strain on the desktop interface, at least in its conventional form (*Moran and Zhai 2007: 340*). Therefore, they believe that alternative or extended forms of information representation guided by different metaphors are necessary. They assume that a variety of advanced visual representations will complement the basic conventional desktop metaphor in future.

Using InfoVis techniques to create such visual representations can greatly improve the efficiency of search and navigation in large information collections. For example, visual information seeking systems such as the *FilmFinder (Ahlberg and Shneiderman 1994)* let users map metadata to the axes of a scatter plot to get a quick overview of an entire database and to recognize clusters of characteristic content. Furthermore, by using dynamic queries with rapid visual feedback, users can quickly formulate complex search and filter criteria for numerical, alpha-numerical, and categorical data to reduce the total amount of information and to focus on a desired subset or individual item (Figure 122). Such InfoVis techniques also enable users to follow Shneiderman's famous *visual information seeking mantra* of *"overview first, zoom and filter, then details-on-demand"* (*Shneiderman 1996*). A further important InfoVis technique, which is however not part of the prototypes in this thesis, is combining different visualizations methods to overcome the shortcomings of single techniques with *linking & brushing* (*Keim 2002*).

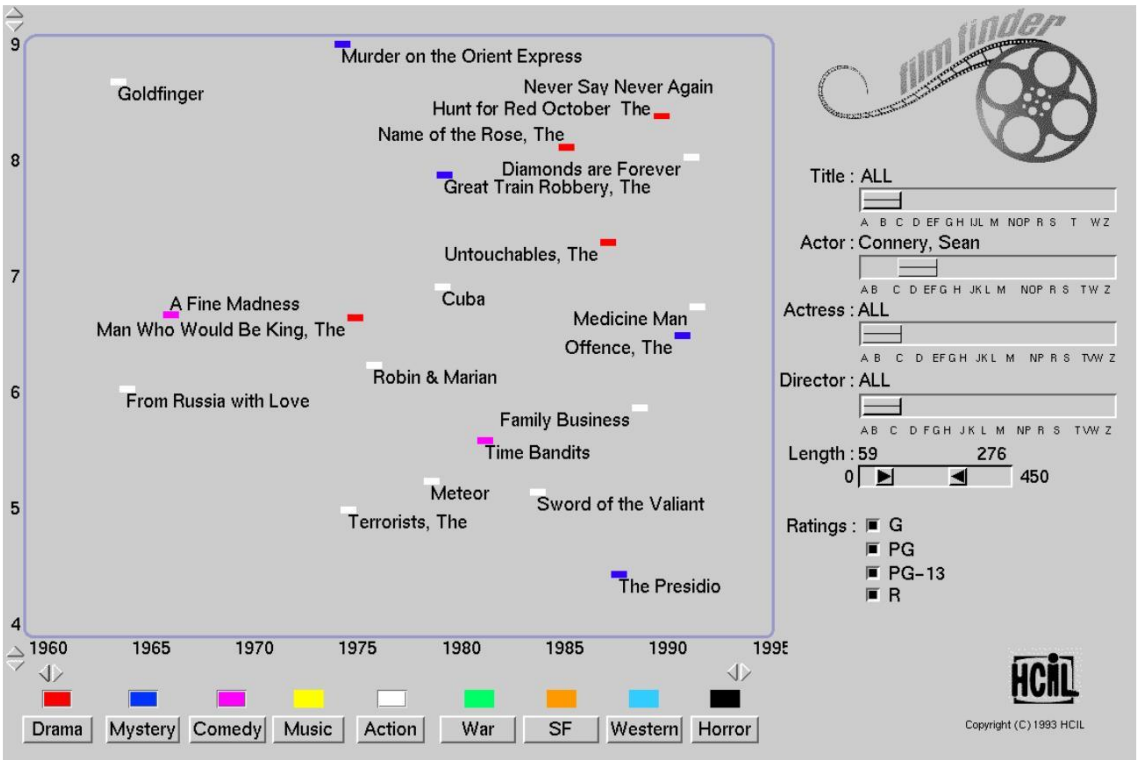


Figure 122 – The *FilmFinder* is perhaps the most famous example of information visualization for the purpose of visual information seeking (*Ahlberg and Shneiderman 1994*).

Figure 123 – The ZOIL-based *HotelBrowser* prototype uses dynamic queries at the top of the screen to filter the hotel objects in a map. Details of the hotels can be accessed using semantic zooming.

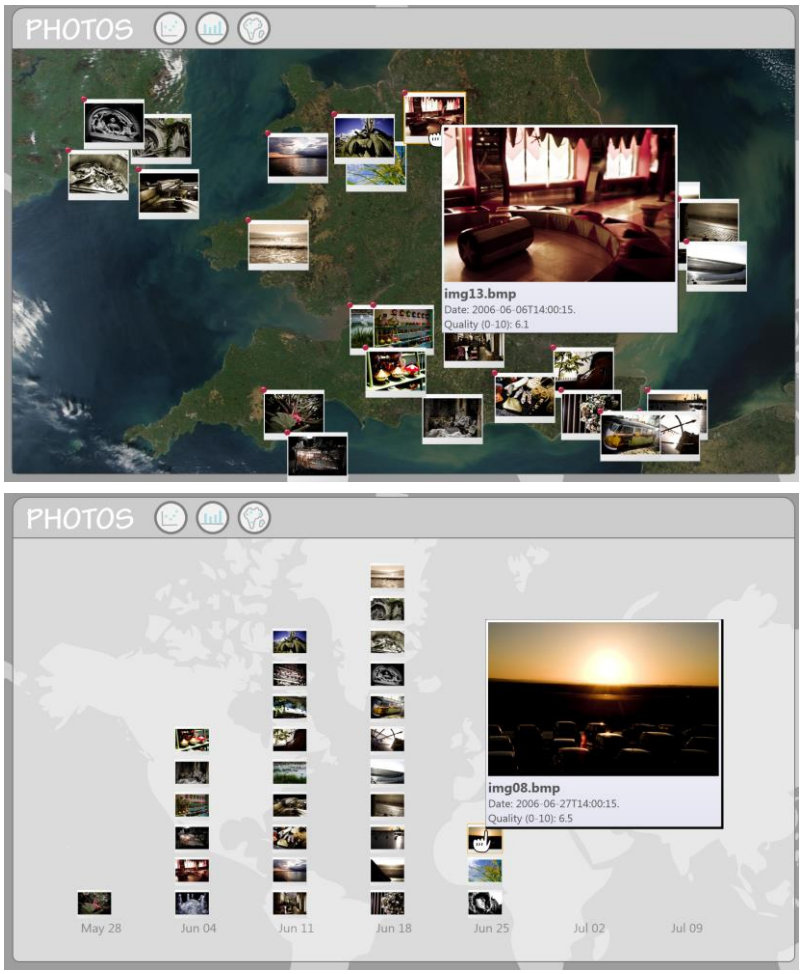


Figure 124 – In the *EuroITV* prototype, users can visualize their photo collection in a map (left) or in a timeline (right).

The ZOIL-based prototypes in this thesis employ a variety of afore-mentioned InfoVis techniques: The *HotelBrowser* prototype applies principles of dynamic queries on a map visualization (Figure 123). The *Media Seminar Room* and *MedioVis 2.0* (*Heilig, Demarmels, Rexhausen et al. 2009*) use free-floating see-through lenses with different visualizations above the information landscape and provide text fields for entering textual filter criteria (see section 4.3.7 and 4.3.8). The default information landscape of the *EuroITV* prototype (*Jetter, Engl, Schubert et al. 2008*) enables users to switch between geographic maps, a time line with weekly stacks, or a scatter plot to visualize their collection of movies, mails, and photos (Figure 124). Although all these examples are simplistic compared to fully-fledged InfoVis systems, they demonstrate the typical benefits of InfoVis and visual information seeking in ZOIL-based interactive spaces.

6.2 Fluid Interaction for Post-WIMP Information Visualization

Information visualization can enhance the user experience beyond the quicker recognition of patterns in data or more efficient information seeking and filtering in document collections. There is a certain class of InfoVis system with particular engaging, compelling, and even absorbing user experiences that go further and turn the analytical sensemaking process into a pleasurable task (*Elmqvist, Vande Moere, Jetter et al. 2011*). This resonates with the increased importance of hard-to-grasp concepts such as user experience, attractiveness, or fun in today's interaction design (*Jetter and Gerken 2006*). In (*Elmqvist, Vande Moere, Jetter et al. 2011*), we therefore propose a unifying concept that captures the properties of this class of InfoVis systems: *fluid interaction*.

Our[59] concept of *fluid interaction* with InfoVis is characterized by a *fluid interface* with one or several of the following properties (*Elmqvist, Vande Moere, Jetter et al. 2011*):

1. The interaction should be designed to promote "*staying in the flow*" (*Csikszentmihalyi 1990*), i.e., it should promote factors such as a balanced challenge, concentration, prompt feedback, sense of control, loss of self-consciousness, and transformation of time.

2. The interface should support *direct manipulation* with continuous representation of the object of interest and physical actions instead of complex syntax. User should be able to apply rapid, incremental, and reversible operations whose impact on the object of interest is immediately visible. This fosters a layered or spiral approach to learning that permits usage with minimal knowledge (*Shneiderman 1983*).

3. The design of the user interface should minimize the distance between the task the user has in mind and the way that task can be accomplished via the interface.

[59] Apart from the description and analysis of *Facet-Streams* in that article (section "4.1 Facet-Streams"), my contribution to it was in particular the introduction of Shneiderman's notion of *direct manipulation* and Hutchins et al.'s cognitive model of the *gulfs of execution* and *evaluation*. By this, I provided a first cognitive account of *fluid interaction* for InfoVis and I consequentially authored also the article's section "3.2 Towards a Cognitive Account of Fluid Interaction" that is partially reproduced here.

> This bridges the *gulfs of execution* and *evaluation*, so that systems feel natural while providing a feeling of *directness* and *direct engagement*. In order to provide such a feeling of *direct engagement*, the interface must provide the user with a world in which to interact. The objects of that world must feel like they are the objects of interest, that one is doing things with them and watching how they react (*Hutchins, Hollan, and Norman 1985*).

From a cognitive perspective, these aforementioned properties of a *fluid interface* for InfoVis share some important commonalities: Fluid interfaces for InfoVis must make the users feel that they are able to directly "touch", manipulate, and control the visualization instead of indirectly conversing with a user interface. Users should get a feeling of immersion, first-personness, and direct engagement with the objects and the visualizations that concern them.

This phenomenon was first described by Hutchins et al. (*Hutchins, Hollan, and Norman 1985*) in their cognitive account of *direct manipulation*. They differentiated between two major metaphors for the nature of human-computer interaction, a *conversation metaphor* and a *model-world metaphor*: in the former, the interface serves as a language for interacting with the world, whereas in the latter, the interface itself is the world which the user can manipulate. For Hutchins et al., model-world interfaces feel natural and create a feeling of directness by minimizing the *gulfs of evaluation* and *execution* (*Hutchins, Hollan, and Norman 1985*) and thereby using less cognitive resources. For achieving *fluidity*, InfoVis should follow this model-world metaphor.

This is also in line with the growing importance of theories of *embodied cognition* in cognitive science and the greater role that our physical and social existence and skills play in post-WIMP human-computer interaction (*Dourish 2004*; *Jacob, Girouard, Hirshfield et al. 2008*; *Klemmer, Hartmann, and Takayama 2006*). New *embodied* or *reality-based* views emphasize that our cognitive abilities co-evolved with our bodily functions to successfully act in our natural physical and social world (*Gibbs 2006*). In other words, our entire way of thinking and our perception are defined by the real-world problems that we encounter in our physical and social environment and by the question how we can purposefully manipulate these environments or successfully partake in them. These problems and questions are far more constituent for human cognition than those skills that are traditionally considered as our cognitive strengths, e.g., processing symbols, abstract thought, and using formal mathematical notations or artificial languages. In reality, we are wasting a great deal of our innate and learned cognitive abilities when using computing technology without directly manipulating objects in (model-)worlds and reducing human-computer interaction to a conversation by typing commands or clicking through sequences of labeled buttons, menus, hyperlinks, or forms. For this reason, today's post-WIMP user interfaces employ a stronger form of direct manipulation than the traditional drag-and-drop of desktop computing with a mouse. They use modalities such as body gestures, multi-touch interaction, or tangible objects to further reduce the

users' *gulf of execution* and to draw strength by employing themes of reality such as body, environment, and social skills & awareness (*Jacob, Girouard, Hirshfield et al. 2008*).

In conclusion, given our innate cognitive abilities and our experiences from interacting with the real world, creating model-world interfaces into which users can immerse themselves and in which objects of concern become virtually or even physically tangible can help to achieve Csikszentmihalyi's factors of flow (*Csikszentmihalyi 1990*): Directly manipulating the objects in the model-world with a greater set of motor skills (e.g., directly dragging an object with multi-touch or using a tangible user interface element to that effect) mediates a sense of control and the model-worlds provide the desired prompt feedback. They use less cognitive resources and thus enable concentration on the task instead of concentration on handling the user interface. Less usage of cognitive resources can also help to achieve a greater design space for a more balanced challenge. Furthermore, good direct manipulation interfaces are in many respects similar to computer games (*Shneiderman 1982*) and thus could lead to the loss of self-consciousness and transformation of time as they are required to achieve "flow" in the sense of Csikszentmihalyi.

6.3 Fluid Interaction with Facet-Streams

In recent HCI and InfoVis literature, our ZOIL-based *Facet-Streams* prototype from section 2.4 is considered an example of successful InfoVis interaction beyond mouse and keyboard (*Lee, Isenberg, Riche et al. 2012*) and future "natural" search user interfaces for social search (*M. A. Hearst 2011*). Furthermore, in (*Elmqvist, Vande Moere, Jetter et al. 2011*), it is used as a *"best-in-class"* example of fluid post-WIMP interaction with information visualization. Therefore, the following sections present the design of *Facet-Streams* and the results and findings of its evaluation with users to derive and illustrate two further ZOIL design principles. These two principles are intended to capture the essentials that make *Facet-Streams* successful, so that ZOIL designers can draw from these experiences. For this purpose, the following sections summarize the original *Facet-Streams* publication (*Jetter, Gerken, Zöllner et al. 2011*)[60] and view it through the lens of the ZOIL paradigm.

6.3.1 Motivation for Facet-Streams

The use of tabletops for co-located collaborative search is an ongoing topic in HCI research (*Morris, Fisher, and Wigdor 2010*). Tabletops can offer diverse benefits and potentials for collaborative search such as a closer face-to-face collaboration and more equitable working style (*Rogers and Lindley 2004*), an increased awareness and better

[60] *Facet-Streams* is the result of joint work with Natasa-Milic Frayling of Microsoft Research Cambridge on collaborative faceted search on tabletops. *Facet-Streams* was conceived of and designed by me together with Jens Gerken. The technical implementation was done by me with support from student developer Michael Zöllner. The user study in the *Facet-Streams* CHI publication that I authored was designed by me and Jens Gerken. Also the data analysis was done by me and Jens Gerken. The practical execution of the user study was supported by Mathias Heilig, Svenja Leifert, and Stephan Huber. The user study of *Facet-Streams* has not been previously published except in (*Jetter, Gerken, Zöllner et al. 2011*).

group work experience (*Amershi and Morris 2008*), and a horizontal form-factor whose affordances are well-suited to follow-up activities, e.g., sorting, sensemaking, making a purchasing decision (*Amershi and Morris 2008*; *Morris, Fisher, and Wigdor 2010*). However, other potentials of tabletops for search are still unexplored, e.g. the use of *"hybrid surfaces"* like (*Jordà, Geiger, Alonso et al. 2007*) that use tangible interaction with physical props in combination with multi-touch (*Kirk, Sellen, Taylor et al. 2009*). Except the ZOIL-based *Search Token* from section 3.5.4, such hybrid tabletop interaction has not been used in search scenarios yet. Furthermore, in the light of the growing popularity of tabletops in showrooms or flagship stores, it is surprising that no prior research has focused on the obvious task of collaborative search for products in a retail environment.

Facet-Streams is a novel system designed for collaborative faceted product search. It uses a hybrid interactive surface that combines information visualization techniques, i.e., a filter/flow metaphor (*Young and Shneiderman 1993*), with tangible and multi-touch interaction to materialize collaborative search on a tabletop. Thereby, unlike in most previous work, the notion of search in *Facet-Streams* does not mean to populate an empty workspace with the results from a keyword search. Instead, it means a process of collaborative faceted filtering of a product catalog until the amount of results is sufficiently small to review and decide (*Yee, Swearingen, Li et al. 2003*). Furthermore, in a retail environment like a flagship store a "good" customer experience with "soft" factors such as fun, innovative design, and social experience is often valued over "hard" factors such as task completion times and error rates. Thus searching with *Facet-Streams* should not be evaluated based only on its objective efficiency, but also by the user experience that users have during collaborative search tasks.

The work on *Facet-Streams* has therefore been guided by three research questions:

(Q1) Does the design turn collaborative product search into a fun and social experience with increased group awareness? (see section 1.4.1 for the definition of awareness and its importance for collaboration).

(Q2) Can we support the great variety of different search strategies and collaboration styles in different teams with a simple but flexible design? (see section 1.4.3 for the discussion of the importance for switching between tightly-coupled collaboration and loosely-coupled parallel work).

(Q3) Can we harness the expressive power of facets and Boolean logic without exposing users to complex formal notations? (also see section 6.2 for the importance of a balanced challenge and minimizing the use of cognitive resources for fluid interaction).

The following sections first discuss related work and the specifics of the context of use. Then the design rationale of *Facet-Streams* is discussed. This is followed by a description of two user studies and a discussion of their results in terms of user experience, collaboration styles, and awareness. This in-depth treatment of Facet-Streams is concluded by summarizing the results and discussing them with respect to the research questions and two new ZOIL design principles.

6.3.2 Related Work

The work on Facet-Streams roots in the state-of-the-art from three fields of HCI research: 1.) tabletop interfaces for collaborative search, 2.) tangible queries, and 3.) visual query languages.

Tabletop Interfaces for Collaborative Search

Morris et al. provide a comprehensive overview of the current research on tabletop search systems and contrast the different approaches using different dimensions such as *search input, collaboration style* and *application domain (Morris, Fisher, and Wigdor 2010)*.

Regarding search input, *Facet-Streams*' design is novel as it is the first approach that employs hybrid surfaces with tangible and touch interaction for search applications. All previous approaches entirely rely on touch, mouse, or keyboard input without making use of any physical props as tangible user interface elements.

Regarding the collaboration style, *Facet-Streams* is similar to *FourBySix Search* (*Hartmann, Morris, Benko et al. 2009*), *Cambiera* (*Isenberg and Fisher 2011*), and *WeSearch* (*Morris, Lombardo, and Wigdor 2010*) which all support seamless transitions between tightly-coupled collaboration and loosely-coupled parallel work. However, unlike these applications, *Facet-Streams* does not use keyword search for Web, document, or multimedia retrieval, but uses a visual and tangible query language for faceted search. Thus *Facet-Streams* shares commonalities with *TeamSearch* that also creates a faceted search experience based on Boolean-style AND queries on tagged photo collections (*Morris, Paepcke, and Winograd 2006*).

Like *TeamSearch*, *Facet-Streams* uses circular widgets to specify categorical criteria (or facets). However, its design goal is to achieve a far greater query expressivity with arbitrary numbers and logical combinations of such widgets including AND and OR. Furthermore, *Facet-Streams* does not restrict a team to only formulate either personal queries or collective queries. Instead it enables them to collaboratively develop multiple queries in parallel and to freely shift criteria between them at any time for maximum flexibility in strategies and collaboration styles.

A further fundamental difference between previous work and our design is the employed notion of search. Except *TeamSearch* and *PDH* (*Shen, Lesh, Vernier et al. 2002*) all systems in (*Morris, Fisher, and Wigdor 2010*) increasingly populate the collaborative workspace with the results of keyword searches. Thereby search has the notion of adding result sets to the shared workspace. In contrast, *Facet-Streams* follows a faceted search approach (*Yee, Swearingen, Li et al. 2003*) where search means narrowing down the entirety of products in the workspace to the desired subset. Thus the focus of collaboration in Facet-Streams lies on the formulation and logical combination of the desired facets of a product, e.g. "price < 100", "WiFi in rooms", *before* reviewing individual results. This is different from related work where collaboration is primarily focused on reviewing and relating of results *after* search. The only systems in (*Morris,*

Fisher, and Wigdor 2010) following a similar faceted approach have either limited expressivity (*TeamSearch*) or force users to only navigate a single facet at a time (*PDH*).

Tangible Queries

In 2003, Ullmer at al. suggested using physically constrained tokens to manipulate database queries and result visualizations in a real estate application (*Ullmer, Ishii, and Jacob 2003*). Two kinds of physical tokens (knobs and range sliders) serve as tangible input controllers that are put into slots next to a display. Although enabling some basic Boolean logic between the range sliders, the overall expressivity is limited to assigning database fields to the axes of a scatter plot and altering parameters of predefined queries. Nonetheless, this design has been the inspiration for a variety of further designs of tangible queries, e.g., for facilitating search for children (*Detken, Martinez, and Schrader 2009*), or Blackwell et al.'s *Query by Argument (QBA)* system (*Blackwell, Stringer, Toye et al. 2004*). *QBA* enables groups to manifest the course of an argument in spatial configurations of "statement tokens", i.e., RFID-tagged cards as physical place-holders for contributions to the discussion. Each token carries a reference to a virtual information item that contains the contribution, e.g., relevant text passages. By spatially configuring the statement tokens during discussion, the group provides continuous relevance feedback to an information retrieval system that tracks and evaluates the spatial structure in the background to adjust its ranking mechanisms. As a result, the system suggests helpful related material on a peripheral screen or projector. *QBA*'s approach to use spatial configurations of tokens to materialize the (chrono-)logical order of an argument during a collaborative process has been inspirational for the use of a network of tokens for faceted search. However, *Facet-Streams* provides users with tokens for precise filtering and immediate feedback. *QBA* is targeted at working invisibly in the background to gradually adjust its ranking without the same need for preciseness and immediacy.

Visual Query Languages

While being of great practical value for search tasks, Boolean AND and OR are difficult concepts to grasp and they contradict the linguistic sense of "and" and "or" in our natural language use (*Borgman 1996*). Therefore, many attempts have been made to move these concepts from a linguistic onto a visual layer of reasoning. For example, today's faceted search on e-commerce Web sites enables users to formulate the equivalent of a sophisticated Boolean query by taking a series of small, simple navigation steps (*M. Hearst 2009*). However, these page-oriented designs for the Web do not suit the scenario of co-located collaborative search around a tabletop in which multiple personal or group queries shall be explored in parallel. Young et al.'s filter/flow metaphor (*Young and Shneiderman 1993*) provides a visual alternative by spreading out a Boolean query into a visual workspace in the form of a chain of criteria for direct manipulation. Logical relations between criteria and remaining results are visualized as streams of different thickness. However, this approach is not directly applicable to tabletops, since its uni-directional layout needs too much screen estate for a tabletop and only permits a single

query per workspace. *FindFlow* (*Hansaki, Shizuki, Misue et al. 2006*) expands the filter/flow metaphor into two dimensional networks with better use of screen estate. However, its concept of displaying entire result sets within nodes conflicts with size restrictions and legibility on tabletops during around-the-table interaction. Furthermore, *FindFlow* does not support the formulation of the logical OR that *Facet-Streams* provides to enable users to specify complex criteria, e.g., “either the hotel has a good restaurant OR I want to have a small kitchen in my room”.

6.3.3 The Design of Facet-Streams

The design of *Facet-Streams* is based on a scenario of use in which three family members gather around a Microsoft Surface in the showroom of a travel agent. They want to find and agree on one hotel for a week of vacation in Europe. Each of the 204 hotels in the catalog carries 12 quantitative or categorical facets, e.g., *price per night, recommendation rate, hotel features, country*. For every hotel each of these 12 facets has a single value, e.g., Hotel A has *price per night* = 50 EUR; *recommendation rate* = 80%; *hotel features* = WiFi+Gym; *country* = Spain. This scenario was used throughout the entire design process and the user studies. To ensure that the contained product data is realistic, a subset of the catalog of a large online travel agency including authentic user photos and ratings was used.

The starting point of the design has been a simple grid layout on ZOIL’s information landscape that contains all hotels (Figure 125). This initial state of the system is the *browsing mode*. In this mode, users can access the details of a hotel by semantic zooming into its icon using typical multi-touch manipulations (tapping or pinching for zooming, sliding for panning). Users are enabled to collaboratively browse a catalog of products at different levels of detail while maintaining spatial orientation and using visual cues or thumbnails for recognition. However, this *browsing mode* is hardly efficient for a search scenario in which various criteria have to be met: While family member A is primarily interested in a close beach, family member B prefers a place with good service and food, and family member C is concerned about the travel budget. Finding a hotel that suits all these criteria by manually reviewing each hotel would be a too tedious and error-prone task.

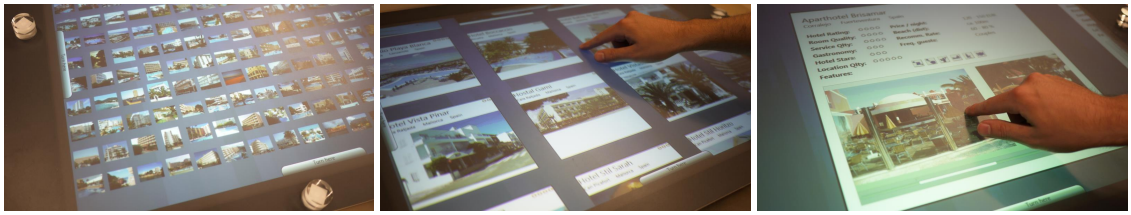


Figure 125 – ZOIL’s information landscape containing all hotels for semantic zooming [see Figure 36 on p.48 for an enlarged version].

Using Physical Tokens for Faceted Search

To leverage the power of collaborative faceted search, *Facet-Streams* enables each user to specify a set of personal criteria (e.g. *country* = Spain and *price* < 120 EUR) for

narrowing down the number of displayed hotels. This is done using physical *facet tokens* (Figure 126). Each token is a circular glass disc of 30 mm x 12 mm. Each token carries a unique rectangular fiduciary marker (19 mm x 19 mm) on its bottom side. The id, X and Y positions, and orientations of all markers are tracked by the tabletop's vision system.

Figure 126 – A *facet token* (left) and a *result token* (right). Source: (*Jetter, Gerken, Zöllner et al. 2011*).

The choice of the form and material of the token is not random, but is intended to create affordances and make appropriate actions perceptible to the user (*Norman 2002*). The token's shape resembles that of a piece on a checkers board or an ice hockey puck and thereby affords sliding the token with the flat side lying on the tabletop. Its diameter invites to comfortably grab the token between thumb and index finger and enables rotational movements of sufficient precision. Thus the token's physical form already hints at its manipulation possibilities of translation and rotation. This also helps to prevent unanticipated uses such as leaning tokens or rolling tokens over the table in an upright position.

The choice of glass has been based on previous observations of issues of hygiene in tangible and touch interaction (*Ryall, Forlines, Shen et al. 2006*): Glass tokens might be considered as more hygienic than plastic or wood by those who feel uncomfortable with touching a shared object. Furthermore, future designs could explore the use of transparent markers printed with infrared-absorbing ink, so that the glass token itself could be used as a small screen for displaying further information.

Figure 127 – A facet token with *hotel stars* = 1 *or* 5 as criterion (left). Touching the current facet ("Hotel Stars") will invoke the *facet wheel* (center). Touching the current value ("1 Stars, 5 Stars") will invoke the *value wheel* (right) [see Figure 37 on p.49 for an enlarged version].

After a *facet token* has been put on the tabletop, the system switches into *query mode*. A new translucent layer appears above the canvas, the canvas freezes, its colors are dimmed, and the screen zooms out to display the entirety of hotels in the background of

the new layer. On the new layer, virtual elements are displayed around each *facet token* to augment them with content and functionality (Figure 127).

Users use a *facet token* to formulate a criterion by selecting a *facet* (e.g. *hotel stars*) and specifying the desired values (e.g. *hotel stars* = 1 or 5). For changing the facet, users touch the currently selected facet that is displayed along the inner circle around the token. This opens a circular pop-up menu called *facet wheel* in which a single facet can be selected by touching its wedge. For changing the desired value range, users touch the currently specified range that is displayed along the outer circle around the token. This opens the *value wheel* in which values can be specified by selecting or deselecting value wedges by touch. The value wheel either carries wedges with values for categorical facets (e.g. CableTV, Restaurant) or quantitative value ranges (e.g. < 50 EUR, 50 - 80 EUR).

Collaboration Styles: Collective vs. Personal Criteria

An obvious approach to formulate queries with the *facet tokens* would be that of *TeamSearch* (*Morris, Paepcke, and Winograd 2006*): By combining the criteria from all *facet tokens* on the tabletop with a logical AND, *Facet-Streams* could provide a simple mechanism to formulate a collective query. However, as mentioned before, the intention behind *Facet-Streams* is to introduce a greater degree of flexibility in search strategies and collaboration styles to support mixed-focus collaboration (*Tang, Tory, Po et al. 2006*), which is of great importance for collaborative search (*Morris, Fisher, and Wigdor 2010*) and in particular in our case of product search.

A collective query based on a logical AND of all criteria affords very tightly-coupled collaborations. Each modification of a single criterion has an immediate effect on the entire result set. This can be disruptive in earlier phases, when users might start their search with unrealistic and conflicting criteria, e.g., a very luxurious hotel for a very low price, because they did not have the chance to individually explore the available facets and products first. The search process can then quickly deteriorate to a random trial-and-error manipulation of criteria to receive non-empty result sets.

As a consequence, *Facet-Streams* must support phases of parallel personal exploration. Even in later stages, such phases of parallel work can support a single user or a couple of users in verifying and altering their contributions to the collective criteria. Thus, in accordance with (*Tang, Tory, Po et al. 2006*), switching between tightly-coupled and loosely-coupled work and flexible coupling between group members is important.

Networks of Facet Tokens as Query Language

As already discussed, *Facet-Streams* employs a fundamentally redesigned version of the visual filter/flow metaphors for Boolean logic of (*Hansaki, Shizuki, Misue et al. 2006*; *Young and Shneiderman 1993*) to enable collaborative faceted search. The first step of redesign was to permit multiple unconnected chains or networks for parallel collaborative exploration. The second step of redesign permitted their use on tabletops by improving the use of screen estate and largely reducing the visual complexity and the amount of text output.

A fundamental concept in the redesign are *streams*. A *stream* visually and logically connects two physical *facet tokens* (Figure 128). Once connected by a stream, tokens will not lose this connection, even when moved to different or distant locations on the tabletop. The stream connects two tokens until one of them is physically removed from the tabletop or the user manually cuts the stream with a touch gesture.

While new to the search domain, similar designs are used on tabletops for live performances of music or video synthesis (*Jordà, Geiger, Alonso et al. 2007*; *Taylor, Izadi, Kirk et al. 2009*). In *Facet-Streams*, streams are directed connections of criteria for faceted search: the source token emits an output stream of filtered hotels that is received as input by another token. The output stream of a token only contains those hotels from its input stream which meet the token's criterion. Thus, if the token's criterion is *country* = Spain, only those hotels from the input stream that are located in Spain will be emitted into the output stream. If a token has no input stream, it is treated as one of many possible sources or starting points in a network and applies its criterion on the entirety of all hotels in the catalog.

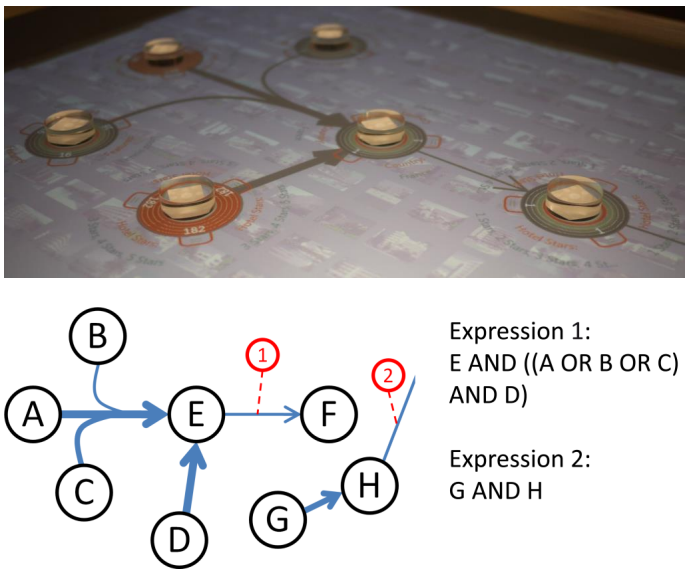


Figure 128 – A network of *streams* that connect *facet tokens* (top). The Boolean equivalent of the networks at ① & ② (bottom). Source: (*Jetter, Gerken, Zöllner et al. 2011*).

Multiple tokens can be connected to chains of criteria where each token only forwards the hotels from the input to the output that match the token's criterion. Therefore, these chains of tokens are an equivalent of a Boolean expression where all criteria are combined using a logical AND. To further extend the expressivity of *Facet-Streams* query language, tokens can have multiple output and input streams. The output streams are all identical and allow users to stream the output towards multiple tokens in parallel. The multiple input streams are combined internally using a logical AND (Figure 128). However, throughout design and user testing, any use of a mathematical or linguistic formalism to convey this Boolean logic on the interface or during instructing users was avoided. The goal has been to move these concepts onto an entirely visual layer of reasoning.

Facet-Streams also provides the possibility to let two streams flow together: By directing an output stream onto an existing target stream, the output stream is bent from a straight line into a Bezier curve that flows into the target stream (Figure 128). Internally the flowing together of two streams is treated as a union of both streams based on a logical OR. The possibility to let streams flow together increases the expressivity of Facet-Streams. The example of a user who wants either a great restaurant or an own kitchen in the room can be covered by letting two according output streams flow together. The resulting union can again be used as input for a further chain or network of criteria.

To better support users' understanding of streams, Facet-Streams provides immediate visual feedback that is updated instantly after any user input: The number of results flowing through a stream is logarithmically mapped to the thickness of a stream. Empty streams are shown as thin lines, but are highlighted in red color to differentiate them from streams with few results. As additional feedback the number of outgoing results from a token is numerically displayed around it. Furthermore, when selecting new value ranges inside a value wheel, each value wedge indicates how many results will remain in the output stream after (de)selection. To get a quick overview of the results flowing in a stream, users can also touch the stream at any time for an immediate query preview. All hotels that are not contained in the stream then disappear from the zoomable canvas in the background and a numerical value with the number of the remaining results appears close to the finger. This kind of feedback can also be achieved by putting a *result token* on a stream (Figure 126). The token additionally provides easy ways to browse sequentially through all contained results or to temporarily switch back into the *browsing mode* for freely exploring the remaining result set.

Designing for Low Viscosity and Parallel Interaction

To enable users to collaboratively explore multiple queries and to shift criteria between them, the design of the lower level interaction techniques of *Facet-Streams* was following the principles of *low viscosity* and *parallel interaction*.

Low viscosity is based on Blackwell et al.'s cognitive dimensions of notations framework (*Blackwell and Green 2003*) and here means a "low resistance to change". This is crucial, since the support for different search strategies and collaboration styles depends on the ability of the users to quickly adapt the topology, the spatial layout, and the criteria of the networks according to the intended working style and goals. For achieving *low viscosity*, the use of a hybrid surface is of great benefit: By coupling the virtual representation of the network to physical props, rearranging the spatial layout is an entirely physical activity without the need for using pointing devices or learning touch gestures. The physical tokens can be relocated by carefully dragging them, but also by carelessly sliding them or even wiping them off the tabletop with the arm. For quick explorations of alternative network topologies, tokens can be lifted from the table and put at arbitrary locations or also into streams. Thus the manipulation of the tokens' locations entirely takes place in the physical not in the virtual world and does not create an uneasy feeling or fear of irreversibly damaging or destroying virtual content.

However, to achieve this, many details of the interaction had to be considered. For example, the selected facet and value range of a *facet token* is stored even after it is removed from the tabletop, so that temporarily lifting or relocating a token does not destroy the contained user settings. Similarly, when removing a token from a network, a set of rules is applied that ensures that all neighboring tokens which have been disconnected by the removal are reconnected in a sensible way. The surrounding network topology is not destroyed.

For other frequent manipulations, *Facet-Streams* uses direct touch instead of tangible interaction: Streams can be created by touching one of the orange handles that are displayed around the tokens (Figure 127). They can then be directed towards their destination with sliding the finger. For cutting streams, a crossing gesture with the finger is used. This direct touch interaction was employed to increase precision and efficiency and also to create a consistent separation of concerns: The position and orientation of the physical tokens are changed using tangible interaction. The creation and cutting of the virtual streams between them happens by direct touch on the virtual elements augmenting the tokens.

The design for *parallel interaction* supports different collaboration styles by enabling users to better parallelize work. For example, a single user might want to modify criteria in a personal network while two other users are working together on a different shared network. Therefore, the interface must be able to handle simultaneous tangible and touch input from all involved widgets such as *facet* or *value wheels* without concurrency issues. Modal interaction or global interface modes must be avoided. Furthermore, all widgets or elements must be movable to quickly establish temporary personal or shared regions on the tabletop, so that parallel user tasks do not interfere with each other. In *Facet-Streams*, all widgets and elements are therefore attached to physical tokens to effortlessly control their position. To ensure accessibility and legibility from all sides, we use a circular design for all widgets to achieve a more equitable interaction without a preferred orientation. Wherever possible, interactive elements or labels around the tokens appear three times every 120 degrees and are curved around the edge of the circular token (Figure 127). A further design for *parallel interaction* is to couple the orientation of a widget to that of its physical token. Thus rotating the token allows effortless changes of orientation without the need to touch virtual handles or to apply touch gestures for rotation. This enables users to quickly show or pass a widget to a collaborator with a different viewing angle. This also enables bimanual interaction of a single user: One hand can be used to rotate the token of a value wheel while the other hand is used for touching the wheel without lifting the finger. This selects the entire value range that passes below.

6.3.4 User Studies

Two user studies with different foci were conducted to get insight about the usability and utility of *Facet-Streams* and to collect empirical data for answering the research questions Q1 to Q3.

Study 1: Comparative Study of Collaborative Use

The first study was intended to observe the use of *Facet-Streams* during a realistic collaborative search task in which a compromise had to be negotiated by the group members. The study should reveal how participants interact with the interface in terms of search strategies, collaboration styles, and parallel interactions. A Web interface for faceted search served as control condition (Figure 129)[61]. Both interfaces were compared in terms of the objective quality of group compromises and the collected qualitative data to contrast the observed personal and group behaviors and strategies.

Design & Participants. The study used a between-subjects design (IV: interface type Facet-Streams or Web interface) with 72 participants, randomly assigned to 24 groups of three. The between-subjects design was chosen because of two aspects that can have a significant and uncontrollable influence in a within-subjects design: First, the novelty of a tabletop with tangibles might evoke a strong "wow"-effect and great bias when put in contrast to a traditional desktop interface. Second, group dynamics evolve over time as people get to know each other. Even a counter-balanced within-subjects design might not be able to rule out interaction effects. The participants were students or faculty from a variety of non-technical subjects (only two students of computer science). The average age was 25 (SD = 7.4 years) with 36 females and 36 males.

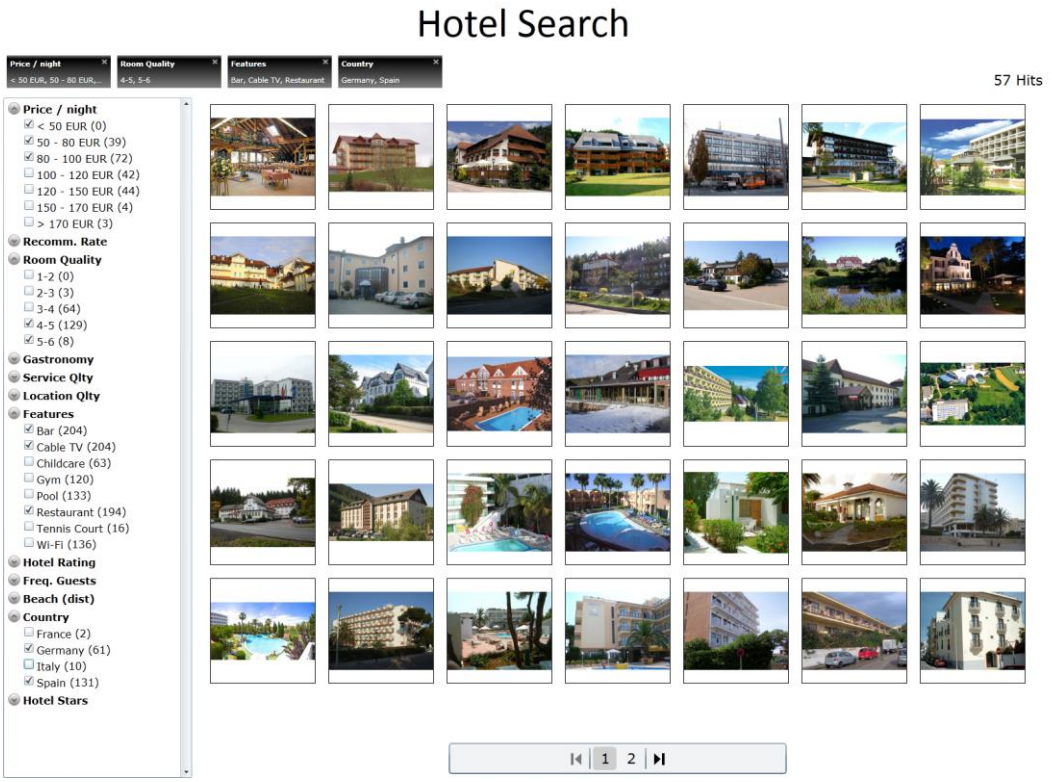


Figure 129 – The Web interface for faceted hotel search. Source: (*Jetter, Gerken, Zöllner et al. 2011*).

[61] A video with *Facet-Streams*, the Web interface, and some footage from the user studies can be found at: *http://www.youtube.com/watch?v=giDF9lKhCLc* (Accessed Jul 29, 2012).

Tasks & Procedure. The tasks required participants to agree on a single hotel from a set of 204 hotels within a limited amount of time. In order to simulate a realistic scenario, participants had to agree on compromises and make concessions to solve a task: For each task every group member was assigned three personal criteria, e.g., “the hotel has to have 3 or 4 stars”, “the distance to the beach must be smaller than 150 m”, “the room quality must be 4 or 5”. However, the task was designed in a way that made it impossible for the group to satisfy the total of nine criteria simultaneously. Simply combining all criteria with a logical AND always led to an empty result set. Thus all participants had to negotiate whose personal criteria to soften, e.g., by extending the price range, or which to give up completely. The whole group was instructed to find an “optimal” compromise that is as close to the entirety of all 9 personal criteria as possible.

Participants were given three of these tasks with varying difficulty in terms of conflicting criteria. A soft time-limit of 7 minutes per task was used to limit and control the session duration and increase participants’ motivation to come to a decision. However, the experimenters did not interrupt users before a final decision was made, since the time limit was not intended as a sharp criterion for task completion or failure. The mean duration of a task has been 6:41 min (mean = 401s, SD = 133s).

Prior to working on the task, each group was given a five minute instruction to the system and five minutes for free exploration. After completing the three tasks, each participant filled out a personal questionnaire about their subjective assessment of the system. Each session took about 45 minutes and was video recorded from different angles to grasp not only the interaction with the interface but also the group dynamics. Participants were compensated with 15 EUR for their time.

Interfaces. The Web interface was created with Microsoft Silverlight to match the state-of-the-art of faceted search on large e-commerce sites, e.g., Amazon.com. The design was slightly more advanced than traditional interfaces since animations and dynamic queries were used to create a more responsive rich internet application. To replicate today’s reality of collaborative search, participants were asked to solve tasks on one 24” screen sharing one keyboard and mouse.

To enable a fair comparison, a pre-test was used to identify specific features from both interfaces that would give users the ability to “cheat” the test design. As a result, the continuous update of the remaining hotels from the checkboxes of the Web interface and from the wedges in the value wheel of *Facet-Streams* was removed. Furthermore, the logical OR from *Facet-Streams* was removed, as the Web interface does not have an equivalent functionality.

Study 1: Results

First, the objective quality of the results that the participants achieved with both systems was analyzed. Therefore, the results for each task from 12 Web interface groups and 11 Facet-Streams groups were compared with the optimal result for each task from the catalog. One group using *Facet-Streams* had to be excluded from the analysis, since one

participant repeatedly ignored the task instructions and used only his personal real-life preferences to judge the group's results.

For analysis, the distance between the given criteria and the selected hotel for each group was determined. The analysis distinguished between concessions and fails. A concession means that the criterion is not met by the hotel, but is met by one of the neighboring values. Each concession adds 1 to the distance. A fail means that the criterion is neither met by the hotel nor by neighboring values, e.g., the given price range was 50-80 EUR and the hotel costs more than 120 EUR. A fail adds 3 to the distance. Deviations in categorical facets such as country were treated as fails. Deviations in features were counted as one concession per missing feature. Figure 130 shows the mean distance for each task. Differences between Facet-Streams and the Web Interface were statistically not significant (T1: t(21)=-0.847, p=0.407; T2: t(21)=0.638, p=0.531; T3: t(21)=-1.517, p=0.144). As all tasks made concessions necessary, the "optimal" value is given as well.

	Facet-Streams	Web Interface	Optimal
T1	2.00 (SD = 1.00)	2.33 (SD = 0.89)	1
T2	6.64 (SD = 1.29)	6.33 (SD = 0.98)	4
T3	3.27 (SD = 0.90)	4.16 (SD = 1.74)	2

Figure 130 – Mean distance and optimal distance for T1-T3.

In comparison to the optimal result, groups achieved good compromises on average with only 1 to 3 concessions or 1 fail per task. Thus both interfaces proved to be effective for the given collaborative search task. This is also confirmed by the subjective assessments of the participants from the questionnaires. On a scale from 1 ('is not true') to 7 ('is true') the mean scores per group for "The system has supported us well for the tasks." are 6.13 (SD = 0.43) for Facet-Streams and 5.86 (SD = 0.77) for the Web interface. While the mean scores are slightly in favor for Facet-Streams this is not statistically significant. Significant differences in favor of Facet-Streams exist concerning the fun users had while using the system and the perceived innovativeness of the design. The mean score for "I had fun working with the system." for Facet-Streams is significantly higher than for the Web interface (6.69 > 5.69, $t(23) = 4.716$; $p < 0.001$). The same is true for "The system is very innovative." (6.38 > 3.61, $t(23) = 8.444$; $p < 0.001$).

Usability and User Experience. It is notable that the users of both systems achieved an equal objective quality of results. Since *Facet-Streams* introduces a novel and unfamiliar style of hybrid tangible and touch interaction with a filter/flow metaphor, it was expected that users would have more difficulties to achieve equally good results within the given timeframe. However, although there was only a brief period of introduction and free exploration, users quickly mastered *Facet-Streams*' interface and did not achieve inferior results compared to the groups using established faceted navigation with a mouse.

Furthermore, as mentioned above, the Facet-Streams interface was reduced in functionality for a fair comparison (no indication of the number of hotels on wedges in the value wheel, no logical OR). In particular the indicators on the wedges would have most likely improved the quality of results in real usage.

With respect to the subjective assessments, participants perceived the design of *Facet-Streams* as something innovative and fun to work with. This is notable, as these assessments were made as part of a between-subjects study to reduce bias resulting from the novelty of a design. This notion of a fun experience was also observed during the sessions: After receiving a new task from the experimenter, one participant turned excitedly to his collaborator and stated *"Nice game!"*. A participant in another group repeatedly stated *"This is so much fun"* and frequently requested more time for *"playing around"*. She was disappointed at the end of the session after realizing that the last task had been solved (*"What a pity!"*).

Search Strategies and Collaboration Styles. Transitions between tightly-coupled collaboration to loosely-coupled parallel work and vice versa were a reoccurring theme in the 11 Facet-Streams groups. From the 33 tasks that were performed by these groups, 26 tasks were begun with a phase of loosely-coupled parallel work. Participants started by building small personal networks of up to 3 tokens in parallel for an initial exploration of their personal criteria. Only after two or more participants had completed this exploration, a phase of more tightly-coupled collaboration with joint networks took place. This hints at the importance of having separate but joinable workspaces.

In the 7 other cases, participants started with a tightly-coupled collaboration from the beginning. Here, all participants sequentially added criteria to a collective chain of tokens. However, in two cases, single participants later seemed to feel that this initial strategy is not meeting their needs. They then started to explore their personal criteria with own tokens in parallel without explicitly discussing this change of strategy with their collaborators. Therefore, the number of tasks that involved loosely-coupled parallel work totals to 28.

During these 28 cases different kinds of transitions from parallel to tightly-coupled work could be witnessed. In 11 cases, all personal networks of the collaborators were merged more or less simultaneously to a single collective query. This was done to identify the hotels that meet all the criteria of the participants and to review candidates that could solve the task. In 12 cases, the transition to collaboration happened between pairs of participants who spontaneously decided to review the intermediate results they share. In 8 cases such pairwise merging of networks was employed systematically to compare results and to identify conflicting criteria: Instead of merging all networks to a collective query, only two networks were merged at a time. Only after the conflicts between each network had been solved by pairwise merging, all networks were merged to a collective query solving the task. It is noteworthy, that merging networks did not interfere with the participants' awareness of which token had been contributed by whom. Whenever a collective network was dissolved into multiple networks again, users easily recognized

their tokens because of their spatial position, the network's topology, and the contained criteria. Thus moving between personal and collective queries was not a challenging task.

Compared to *Facet-Streams*, the search strategies of the Web interface users appeared much less systematic. Of the 36 tasks that were performed, 21 did not follow any recognizable strategy. In these cases, participants shared their personal criteria verbally with the person who was operating the mouse and who collected and entered all criteria in a random order. In 11 cases, the operator used the list of available facets on the screen to sequentially inquire at every facet whether group members had a relevant criterion or not. In the remaining 4 cases, a similar sequential inquiry took place using the criteria cards that had been handed to the participants at the beginning of the task.

Typically, the groups encountered a small or empty result set after few entered criteria. From then on the groups started to modify or soften their collective criteria during tightly-coupled collaboration. Not surprisingly, there was intense verbal communication between the operator and the other two participants during all phases, but in particular during this refinement phase. However, in contrast to the *Facet-Streams* groups, most of this communication happened to check and confirm which criteria had already been entered and which not. Although the region of the Web interface with the checked or unchecked criteria was visible to all collaborators, the groups quickly lost track of the number of entered criteria. Furthermore the participants could not attribute the different checkboxes to individual persons and seemed to be less aware of their own and others' criteria. This led to a lot of "noise" in the verbal communication that primarily served to create awareness for the current system state, but seldom for suggesting actual steps for problem solving. Not surprisingly, this also led to a more browsing-oriented search strategy. Compared to Facet-Streams, manual reviewing of results happened earlier and lasted longer.

In conclusion, the interaction with the Web interface suffered from a lack of awareness and from the alienation of the participants from their criteria after being entered into the system. This is contrary to our observations for *Facet-Streams* and hints at the role that the tabletop plays as a space for creating awareness.

Discussion. Hornecker & Buur introduce a framework on physical space and social interaction for tangible user interfaces. They emphasize the importance of *externalizations*, e.g., a shared visual or physical workspace, to support communication, negotiation, and shared understanding. Externalizations can aid cognition, provide shared reference, and remember our traces. They can directly or indirectly foster collaboration and awareness (*Hornecker and Buur 2006*).

However, it is necessary to design such externalizations with care. Own work has revealed that touch-sensitivity can interfere with natural non-verbal communication in which deictic gestures play an important role (*Gerken, Jetter, and Schmidt 2010*): Touching objects is not necessarily an explicit input to a system, but can instead serve the purpose of indicating locations or simulating movements during explanations. In

these cases, touch-sensitivity can lead to unintended touch interaction and result in a less precise communication or higher physical strain.

In *Facet-Streams*, the shared workspace on the tabletop with networks of tokens served as a successful visual-tangible externalization. The networks provided a visual and tangible map of the search process that captured its (chrono-)logical development. Users were able to attribute tokens and their criteria to other collaborators based on the topology and spatial distribution of the networks. They were able to interpret the networks as an indicator of their progress in search and used them to store, revisit, and reuse intermediate results.

These features of the tabletop led to an increased *awareness* that became obvious at several occasions: Frequently, participants reminded each other of their respective criteria. For example, one participant reminded her collaborator to think of the collaborator's children, as the collaborator was about to abandon the requested hotel feature "childcare". Increased awareness also became visible during phases of loosely-coupled parallel work: Although being busy with his own personal network, one participant realized that another collaborator had problems handling the value wheel and quickly interrupted his own work to help out. In some cases, such mutual support involved verbal communication and pointing towards the elements to touch or turn. In other cases, it happened without any prior verbal communication or coordination. For example a participant noticed that one of his collaborators had repeatedly tapped a hotel to review its content. However, the system was currently in *query mode* that does not support this feature. Without any further communication this participant grabbed the close-by result token to switch to *browsing mode*, so that his collaborator's flow of work was not interrupted. Such interactions can be regarded as first steps towards creating a "flow" experience within a group and an example of fluidity in interaction with InfoVis.

In contrast, the Web interface's externalization without spatial or topological cues failed to provide this degree of awareness. Users were also less able or more reluctant to provide mutual help. In one case, the operator of the interface made frequent mistakes while (un)checking criteria. However, the two collaborators who closely watched his interactions did not realize these mistakes or decided not to intervene.

Observed Problems. While the *query mode* was well-received, usability flaws were observed during *browsing mode*. For browsing, most groups gathered at a single side of the tabletop, since the orientation of the results could not be controlled individually. In future, users must be enabled to read, arrange, and pile results from all sides for an equitable collaborative browsing. Furthermore, users often had to move tokens out of the tabletop's center, since they occluded results or felt too distracting. Finally, the division in *query* and *browsing mode* led to confusion because the current mode and its available functions were not indicated clearly. In future designs, occlusion should be minimized and different modes should be avoided by merging their functionality.

Study 2: Comprehensibility of Filter/Flow Metaphor

A second study was conducted to evaluate the comprehensibility of *Facet-Streams*' metaphor for Boolean logic. Seven participants were invited to single user sessions (4 male, 3 female, avg. age 23.7 years, SD = 2.75). Participants with a background in computer science or mathematics were excluded. The *Facet-Streams* interface used in this study included the entire functionality described in the design section, including all numerical feedback and Boolean OR.

Tasks & Procedure. Participants were briefly introduced to the interface for 5 min and had further 5 min to explore it. Then a single task from the first study was presented as a "warm-up" task. In the following task 1, they were presented with pre-defined networks of facet tokens on the tabletop to the participants. Each network was manually set up by the experimenter without any explanation. To analyze the participants' understanding of the network, they were handed a printed index card of a hotel with its values for different facets. Participants then had to answer for four different locations in the network whether this hotel was contained in the stream flowing there (Figure 7). Thereby participants were not allowed to alter the network at any time. They were only told whether their answer was correct or wrong without any further explanation. This was repeated with hotels and networks of increasing difficulty.

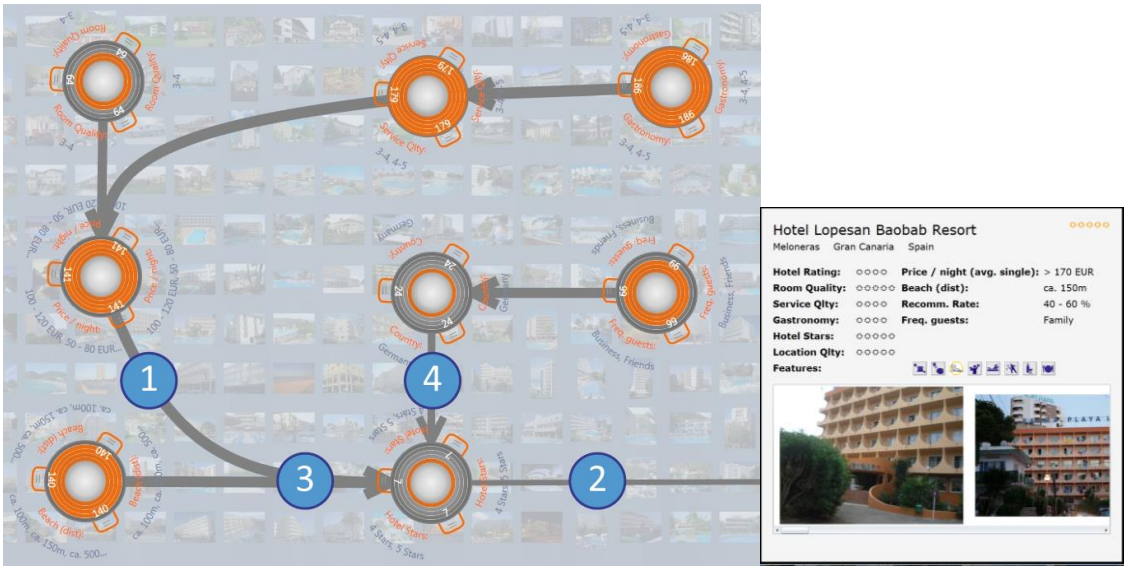


Figure 131 – Left: Network from task 1 with question locations ① – ④.
Right: Example of a printed index card of a hotel that from task 1.

Each participant answered 3 (hotels) x 3 (networks) x 4 (locations in the network) = 36 questions. While instructing participants, experimenters strictly avoided any terminology from Boolean logic or set theory, so that the visualization had to "speak" for itself. This was done to reduce a potential bias based on prior knowledge of Boolean logic or mathematics.

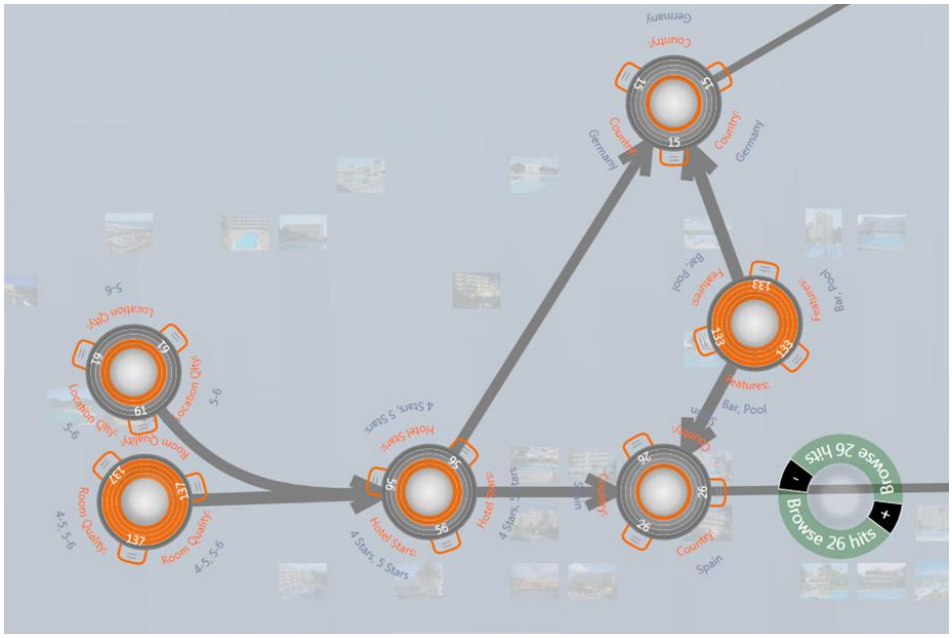


Figure 132 – The correct solution to task 2 after all 7 steps.

Task 2 observed whether participants were able to create complex networks from natural language input. The experimenter played the role of a customer of a travel agency and the participant was asked to use the system to construct a network that answers the customer's questions. The experimenter increased the complexity of his requests in 7 steps following the course of a realistic narration in natural language without referring to concepts from Boolean logic or set theory. To solve this task, participants had to add or remove tokens and to create AND and OR connections between them. Furthermore, they were confronted with conflicting criteria that led to zero results and advanced features had to be used, e.g. multiple output streams from one token. Task 2's complexity at step 7 becomes evident in its Boolean equivalent: (*room quality* = 4-6 OR *location quality* = 5-6) AND (*hotel stars* = 4-5) AND (*country* = Germany OR *country* = Spain) AND (*features* = Bar+Pool). Figure 132 shows the visual-tangible equivalent.

Eventually, participants filled out a user satisfaction questionnaire with semantic-differential adjective pairs and were given a compensation of 7 EUR. Sessions took 30 minutes on average.

Study 2: Results.

For task 1 the total number of incorrect answers was 18 of 252 (7.1%). Regarding the subset of questions that involved a logical OR connection, the number of incorrect answers was 14 of 147 (9.5%). The few incorrect answers show that *Facet-Streams* succeeds to provide a learnable visual metaphor that conveyed Boolean logic even without user interaction. It was learned by the participants without any interaction or extensive periods of training. This was also true for the more complex networks including AND and OR. During task 2, 6 out of 7 participants managed to correctly construct the entire network including the logical OR from natural language. This is an

important insight for co-located collaboration where verbal communication and coordination plays a critical role. Furthermore, task 2 also specifically showed the low viscosity of the interface, as participants frequently lifted and rearranged tokens to try out different alternatives.

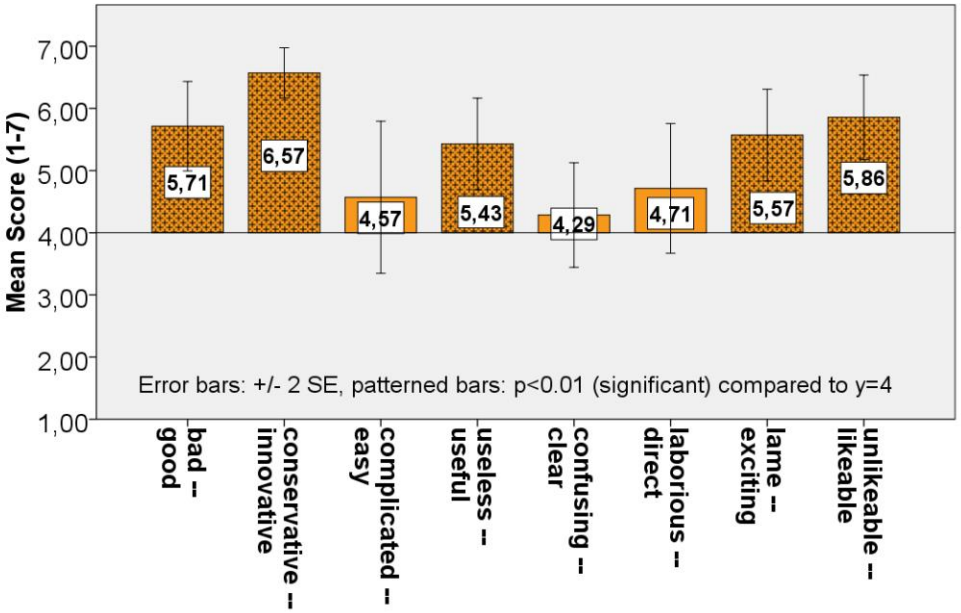


Figure 133 – Semantic differentials from the questionnaires. Source: (*Jetter, Gerken, Zöllner et al. 2011*).

Figure 133 gives an overview of the results of the questionnaires. The scores support our observation from study 1 that Facet-Streams is perceived as appealing, innovative, exciting, likeable, and useful.

6.3.5 Conclusion

Facet-Streams proved to be equally effective as established designs for faceted navigation on the Web, although it introduces novel and unfamiliar hybrid interaction techniques and visual metaphors.

- With respect to research question Q1, users perceived using Facet-Streams as a fun experience and considered its design as innovative. An increased awareness and better mutual support among collaborators was observed.
- With respect to research question Q2, a great variety of search strategies and collaboration styles can be realized with Facet-Streams. This includes seamless transitions between tightly-coupled collaboration and loosely-coupled parallel work.
- With respect to research question Q3, the effectiveness during study 1 and the small failure rates in study 2 confirm that users were able to quickly learn and apply the visual-tangible metaphor for Boolean logic. Thereby they also succeeded in formulating complex Boolean queries based on natural language instructions.

Future work should address the observed usability problems during *browsing mode* and a closer integration of query formulation and result browsing to improve *Facet-Streams'* support for collaborative sensemaking and evaluation of search results. One possible direction for future designs is to integrate further interactive surfaces or mobile displays into the environment to increase display space (see Figure 35, p. 47) and to reduce occlusion by tangible UI elements.

6.4 P#5: "Provide Post-WIMP InfoVis Tools for Fluid Interaction."

This section formulates the *5th ZOIL design principle* based on the considerations about the benefit of InfoVis and fluid interaction and extrapolates from the experiences made with *Facet-Streams* and other ZOIL-based prototypes with InfoVis techniques. It formulates the *5th ZOIL design principle* as follows:

"Provide post-WIMP InfoVis tools for fluid interaction."

This includes following considerations about the benefits of post-WIMP InfoVis and fluidity:

1.) *Information visualization* or *InfoVis* uses computers to create interactive visualizations of abstract data such as numerical values, geolocations, networks, or hierarchies to amplify cognition. This amplification is based on the users' visual perception that enables them to see, explore, and understand large amounts of information at once. If abstract information visualization is interactive enough to change the data transformations, visual mappings, and view transformations, it has the power to reveal patterns, clusters, gaps, or outliers in computer directories, or document collections (*Card, Mackinlay, and Shneiderman 1999*; *Card 2008*; *Shneiderman 1996*; *Thomas and Cook 2005: 30*).

2.) InfoVis systems for visual information seeking system use different forms of information representations, e.g., maps, timelines, scatterplots, to enable users to navigate and search in large information collections. In ZOIL, such representations can be integrated directly into the information landscape (see *EuroITV*) or they are provided in virtual or physical see-through lenses (see *Media Seminar Room* or *MedioVis 2.0*). They can be combined with dynamic queries or other search and filter mechanisms to enable visual information seeking.

3.) InfoVis can enhance the user experience with particular engaging, compelling, and even absorbing user experiences that turn the analytical sensemaking process into a pleasurable task (*Elmqvist, Vande Moere, Jetter et al. 2011*). Ideally, they achieve a *fluid interaction* based on directly interacting in a model-world instead of conversing with an InfoVis interface. Such model-world interfaces promote "staying in the flow" (*Csikszentmihalyi 1990*), support *direct manipulation* (*Shneiderman 1983*), and bridge the *gulfs of execution* and *evaluation* (*Hutchins, Hollan, and Norman 1985*).

4.) Today's post-WIMP InfoVis user interfaces employ a stronger form of direct manipulation than the traditional drag-and-drop of desktop computing with a mouse

(*Elmqvist, Vande Moere, Jetter et al. 2011*). They use modalities such as body gestures, multi-touch interaction, or tangible objects to make use of our full range of innate or learned cognitive abilities such as body, environment, and social skills & awareness (*Jacob, Girouard, Hirshfield et al. 2008*). Using post-WIMP interaction is therefore a particularly promising approach for achieving fluid interaction with InfoVis.

5.) The presented user studies of the post-WIMP *Facet-Streams* system reveal how a visual-tangible Filter/Flow metaphor enables users to quickly learn and apply faceted search and Boolean logic. Thereby the users succeeded in translating natural language instructions in complex Boolean queries. This demonstrates that abstract data and logic can be materialized on the user interface to make them easier to understand and to manipulate for the users. As a result, the users perceived the system's design as appealing, innovative, exciting, likeable, and useful.

6.5 P#6: "Support Multi-User Collaboration with Visual-Tangible Externalizations."

Similar to the previous section, this section formulates the 6^{th} *ZOIL design principle* based on the experiences made with *Facet-Streams*:

"Support multi-user collaboration with visual-tangible externalizations."

This includes following considerations about the benefits and challenges of using visual-tangible externalizations:

1.) In *Facet-Streams*, the shared workspace on the tabletop with visual-tangible networks of personal and collective search criteria serves as a successful visual-tangible externalization of collaboration. It provides a map of the search process that captures its (chrono-)logical development. Users used it as an indicator of their search progress and to store, revisit, and reuse intermediate results. This led to an increased awareness and to close collaboration that showed qualities of a fluid interaction with first steps towards a collaborative "flow" experience.

The success of using a virtual network of physical tokens resonates with (*Hornecker and Buur 2006*) and the role of tangible user interfaces as *externalizations* that support communication, negotiation, and shared understanding. Externalizations can aid cognition, provide shared reference, and remember traces. They can directly or indirectly foster collaboration and awareness (*Hornecker and Buur 2006*).

2.) As demanded by (*Tang, Tory, Po et al. 2006*), *Facet-Streams*' visual-tangible externalization succeeds in supporting transitions between tightly-coupled collaboration to loosely-coupled parallel work and vice versa. Moving back and forth between personal and collective queries is not a challenging task. Merging or splitting networks does not interfere with the participants' awareness for their tokens. Users easily differentiate their tokens from that of others based on their spatial position, the network's topology, and the contained criteria.

3.) It is necessary to design visual-tangible externalizations with care. In *Facet-Streams*, the choice of the form and material of the tokens is not random, but is intended to create affordances and make appropriate actions perceptible to the user and to prevent unanticipated uses. For achieving *low viscosity*, the virtual representation of the network is coupled to physical props, so that rearranging the spatial layout or adding and removing tokens is an entirely physical activity without further learning effort. For supporting *parallel interaction*, the interface must be able to handle simultaneous tangible and touch input from multiple users without concurrency issues. Furthermore, modal interaction or global interface modes must be avoided and the design of controls must provide accessibility and legibility from all sides.

4.) The user study of *Facet-Streams* still revealed usability problems that future designs of visual-tangible externalizations should avoid. Users often had to move tokens out of the tabletop's center, since they occluded results or felt too distracting. In future, occlusion by tangible elements should be minimized, for example by using transparent fiducial markers with IR-absorbing ink, using greater tabletops to increase the visible region, or integrating further interactive surfaces or mobile devices for browsing and sensemaking into the environment.

7 Conclusions & Future Work

This chapter concludes this thesis by summarizing its contributions, discussing limitations, and giving an outlook on future work. In a first step, I therefore revisit the research question for this thesis that I formulated in the introduction:

> *How can designers and developers of ubiquitous computing environments be supported to create more usable multi-user and multi-device post-WIMP interactive spaces for co-located collaborative knowledge work?*

As discussed in the introduction, the answer this thesis gives to this question is the *ZOIL technological paradigm* (see section 1.6.3, Figure 9, p.23) that consists of three components:

1.) The six *ZOIL design principles* that define ZOIL's interaction style and provide "golden rules" to support interaction designers (see section 1.6.1).

2.) The *ZOIL software framework* that supports developers during the implementation of post-WIMP interactive spaces for collaborative knowledge work and the realization of ZOIL's design principles in practice (see section 1.6.2).

3.) The four *example prototypes* based on ZOIL from chapter 2 that can serve as exemplars for designers and developers likewise (see section 1.6.3).

In the following, I summarize the different contributions of the first two components to present the different design, technological, and scientific contributions of this thesis. I also discuss their limitations and suggest possible next steps for future work.

The contributions of the third component, i.e., the four example prototypes, are not discussed in detail here, because they primarily served the first and second components as exemplars, illustrations, and systems for user studies. However, as discussed in detail in sections 7.1.5 and 7.1.6, the *Facet-Streams* prototype sticks out as a self-sufficient scientific contribution with an own publication in (*Jetter, Gerken, Zöllner et al. 2011*).

7.1 The ZOIL Design Principles

The first component of the ZOIL paradigm are the six ZOIL design principles whose contributions I summarize in the following.

7.1.1 P#1: "Provide Post-WIMP Functionality as Objects, not Applications."

The 1st ZOIL design principle introduces a new interaction model for collaborative post-WIMP interactive spaces. This interaction model departs from the application-orientation of traditional WIMP interfaces and the desktop metaphor. It dissolves

application boundaries by attaching functionality directly to domain objects instead of relying on many different monolithic applications with competing metaphors, interaction models, and file formats. All objects and their functionality co-exist in a single integrated work environment following a consistent interaction model that is based on direct manipulation of objects and "object-action" syntax for accessing the functions attached to the objects' views or menus. ZOIL's object-oriented interaction model makes the domain objects and not applications the first-level citizen of the UI. In ZOIL, there are no identifiable applications, application windows, or application boundaries. This resonates with present-day design principles for natural user interfaces that recommend to focus UIs on their content and to use only a minimum of administrative controls or system states.

ZOIL's interaction model is based on the assumption that the object-oriented meta-model, i.e., thinking about the world in terms of objects, their classes, their properties, their functions, and their mutual relations, is a natural and very efficient way of conceptualizing our environment and is close to the way we reason in our everyday world. As a consequence, object-oriented analysis and design is not exclusive to object-oriented programming, but can also be used to understand application domains and to design appropriate UIs. The rigor of object-oriented analysis and design can be applied to identify the relevant UI objects and to design their views, functionality, and interactive behaviors. Thereby object-oriented modeling techniques such as polymorphism and inheritance can be used to carefully model commonalities and differences between object classes in the class hierarchy. If properly applied, this leads to a coherent and consistent behavior throughout the UI that makes it easier for users to apply previous experiences for discovering new functionality and taking out new kinds of tasks. In the light of ZOIL's application domain, i.e., collaborative knowledge work, object-oriented instead of application-oriented UIs also reflect the inherently object-oriented nature of information items.

A further advantage is that object-oriented UIs are not bound to a rigid application structure but have a flexible structure-by object. This supports the embodied and situated interactions of users during post-WIMP co-located collaboration that do not follow predefined plans or scripts but have to be understood as an ongoing, improvised activity with moment-by-moment responses to the situations in which the users find themselves.

My formulation of this interaction model as ZOIL's 1st design principle in chapter 3 is based on an extensive discussion and summary of literature on object-oriented user interfaces from the era of the WIMP GUI from the 1990s. It revisits these largely forgotten concepts and discusses them in the light of literature about the design of natural user interfaces, personal information management, domain-driven software engineering, and situated and embodied interactions. I therefore consider its formulation as the first scientific contribution of chapter 3 that is based on theoretical discussion and logical deduction from the existing body of literature. Consequentially, it

has been part of previous publications (*Jetter, König, Gerken et al. 2008*; *Jetter, Gerken, Zöllner et al. 2010b*; *Jetter, Zöllner, and Reiterer 2011*; *Jetter, Zöllner, Gerken et al. 2012*).

In addition to this theoretical discussion, the study with student designers and developers from section 3.2 provides an empirical basis for the 1st ZOIL design principle. It revealed that object-oriented analysis and design can be successfully applied to design post-WIMP user interfaces that go beyond traditional GUI interaction techniques and make use of zoomable user interfaces with semantic zooming and information visualization. Although the method first appeared unfamiliar to designers and developers and some training was necessary to teach "thinking in objects" for interaction design, the method was overall considered useful by the participants. I consider the results of this empirical study as the second scientific contribution of chapter 3 that has previously appeared in (*Jetter, Gerken, Zöllner et al. 2010b*).

Limitations and Future Work

While the study from chapter 3 and the existing ZOIL examples prototypes from chapter 2 illustrate that ZOIL's object-oriented interaction model can be put into practice, the empirical basis is still limited. In future work, a comparable study with a larger scale project and more experienced designers and developers could reveal how object-oriented analysis and design for post-WIMP interaction can be applied in a professional and not only educational environment. This study should also include more complex requirements for the UI to build, e.g., including different states of domain objects or sequential domain processes that cannot be expressed in class hierarchies alone.

A further opportunity for future work is the extension of the theoretical discussion of object-oriented interaction to tangible objects. Chapter 3 does not differentiate between virtual objects like in ZOIL's information landscape and tangible objects like in *Facet-Streams*. However, as I write in (*Jetter, Zöllner, Gerken et al. 2012*), object-oriented analysis and design could be a tool that informs designers when deciding on a virtual, tangible, or hybrid representation by looking at the object's class hierarchy, properties, functions, and the cardinalities of its relations.

7.1.2 P#2: "Provide a ZUI for Navigation with Semantic Zooming."

The 2nd ZOIL design principle extends ZOIL's object-oriented interaction model by introducing the zoomable information landscape as shared spatial workspace for organizing and navigating objects in space and scale. Instead of a file system with folders and file names, ZOIL organizes information and tools in the landscape's spatial structure that is not a part of the interface but a part of the content. By using a ZUI, ZOIL avoids using overly concrete real-world metaphors while still exploiting users' familiarity with fundamental concepts and mechanisms from real-world experience, e.g., movement in space and the persistence of objects. Thereby ZOIL taps into our natural spatial and geographic ways of thinking.

Based on existing work on ZUIs, chapter 4 also integrates the established ZUI interaction concepts of semantic zooming, portals, and lenses into ZOIL. The different design

examples from chapter 4 illustrate how these concepts can be used to provide meta-data visualization tools (see section 4.3.8), persist results of filtering or search (see section 4.3.7), control overview+detail views across device boundaries (see section 4.3.8), and create and edit virtual objects with digital pen and paper (see section 4.3.8). It also suggests the new method of 'parallax zooming' to avoid desert fog (see section 4.3.3). Furthermore, it suggests using non-linear space-time functions for ZUI animations and differs from existing literature by recommending a hybrid of fixed zooming speeds and constant ZUI animation durations (see section 4.2.3). Some of these design examples and recommendations have been part of previous publications (*Jetter, Gerken, Zöllner et al. 2010b*; *Jetter, Zöllner, and Reiterer 2011*; *Jetter, Zöllner, Gerken et al. 2012*).

The core scientific contributions of this principle and chapter 4 is the user study in section 4.5 that was published in (*Jetter, Leifert, Gerken et al. 2012*):

Inspired by the emerging view of embodied cognition in cognitive science, it is the first study in HCI that observes the effect of mouse vs. multi-touch navigation on users' navigation performance and spatial memory performance when using panning-only and zooming & panning user interfaces. To achieve this, it also introduces an information-based metric that calculates the 'cost' for navigation in ZUIs.

For panning UIs with touch input, a significant increase of spatial memory performance and navigation performance compared to mouse input was observed. For ZUIs, there was no such increase, although there was significant improvement in task completion times and unanimous user preference for touch. I explained this effect based on a multi-modal sensation of distance when using touch. While panning UIs can benefit from this sensation, this is not the case for ZUIs due to their changing scale factors.

For ZUIs with a mouse, we even observed a better navigation performance in terms of path lengths compared to multi-touch. We can attribute this to frequent problems that participants had with two-finger zoom or pinching gestures when zooming into targets that were close to the screen's edges. This resulted in an inefficient navigation style with more panning. Based on these results, I suggested first design implications to guide interaction designers of multi-touch ZUIs.

Limitations and Future Work

To broaden our understanding of this effect, I suggest bi-variate studies with mouse vs. touch and different screen sizes as independent variables to contribute to the formulation of a predictive quantitative model of the observed effects (see section 4.5.8). In particular, further research on smaller screen sizes or touch pads would help to validate our findings. A follow-up study could also help to shed light on the different nature of the curves for navigation performance that seemed to be parallel for panning and asymptotic for touch.

More generally, I would like to motivate other researchers to conduct studies to understand the effects of Embodied Cognition on HCI. Cognitive science provides strong evidence for the effects that body movement inevitably has on cognitive functions such

as memory but also language use, social behavior, and emotional attachment to other people or things. HCI could strongly benefit from understanding and exploiting these effects when designing future user interfaces and gesture sets for gestural and touch interaction in post-WIMP interactive spaces.

7.1.3 P#3: "Provide Space for Sensemaking, Marks, and Annotations."

The 3rd ZOIL design principle acknowledges the fact that space is an integral part of human cognition and a key resource for collaborative knowledge work. ZOIL UIs should provide users with the freedom to flexibly arrange, cluster, configure, and annotate content in space and scale instead of relying on static and restrictive spatial layouts. My formulation of this design principle in chapter 5 is based on a discussion and summary of literature on user studies of the role of space and annotations from cognitive science, knowledge work, and sensemaking. I therefore consider its formulation as the first scientific contribution of chapter 5 that is based on theoretical discussion and logical deduction from the existing body of literature. In particular, the third ZOIL design principle is based on following considerations:

- Space can serve as a temporary holding pattern for potentially interesting inputs and ideas which cannot be categorized and for which there is no decision how to use them yet. Users must not be forced to file or classify immediately, but should be enabled to establish zones and clusters as rough spatial categorizations for experimentation and reflection.
- Space enables experimental exploration without imposing a strict interpretation. The meaning of spatial relationships can evolve with understanding, so that at the start of the sensemaking process, there is no need for a premature choice of an inappropriate and restrictive organizational structure. UIs should actively support the transition from these initially freeform and loosely structured arrangements to more formal structures during sensemaking. This demands a considered choice of layouts and interaction techniques that are good tradeoffs between entirely free arrangements and restrictive structures.
- Space is not only a medium for establishing spatial relationships between items, but it can also carry extra information, for example, annotations, marks, highlights, or visual links. The appearance of such meta-level information should be captured and reproduced by the user interface to provide context, cue the re-call of an earlier mindset, and aid recognition of items.

These theoretical benefits of space are empirically validated and extended by the findings and design implications from the user study of *DeskPiles* in the NanoPhotonics Centre of the University of Cambridge (see section 5.7.5). In this study, four researchers working in the field of nanophotonics performed realistic tasks of knowledge crystallization including the use of sensemaking, marks, and annotations. The study and following findings are the second scientific contribution of chapter 5:

- The process of spatially laying out, arranging, and rearranging items helped participants to clarify their thinking. Spatial groupings of items helped to gain overviews that would not have been able to achieve using non-spatial linear forms like text documents or slideshows.
- Interaction techniques for spatially arranging items in a ZUI (e.g., snap-to-grid mechanisms, magnetism) can be helpful in initial phases. However, the majority of participants found them constraining and wished to arrange items in a more organic and freeform space. This confirms the demand for a considered choice of layouts and interaction techniques that are good tradeoffs between entirely free arrangements that can be difficult to manage and more restrictive but easy-to-use structures.
- When importing items into spatial layouts, users do not always intend to include entire documents, media objects, or other large packages of information. In many cases, they want to use only smaller parts (e.g., images, figures) as self-sufficient pieces of information or as *pars pro toto* representations for an entire class of objects. When importing items, user interfaces should offer the rapid selection and extraction of such smaller parts.
- Spatial UIs for collaborative knowledge work should provide tools for marking and highlighting. They do not only serve to add meta-level information and context (e.g., marking unclear or particular important content) but also enable simple collaborative visual analysis of data by drawing into diagrams and figures to highlight differences between expectations and data.
- Spatial UIs for collaborative knowledge work should enable users to quickly create note objects as annotations. They are important for representing abstract concepts, new ideas, or resources that do not exist as information items yet and have to be quickly generated during collaboration. They are also important for labeling or providing context and other meta-level information to make the resulting information landscape more comprehensible to others.
- Spatial UIs for collaborative knowledge work should provide a rich linking language for creating visual links between items, including directional links, colored links, and links of different thickness to indicate critical and non-critical paths. Furthermore, introducing different categories of links whose visibility can be turned on and off could help to reduce visual clutter when the network of links becomes dense.

Limitations and Future Work

A limitation of the user study is the small number of participants and that the available functionality and snap-to-grid mechanism in the *DeskPiles* prototype has influenced participants in the ways that they used space, marks, and annotations. A future study should aim at a greater number of participants and providing a more freeform space to enable more organic layouts of information items and more flexible use of space.

7.1.4 P#4: "Provide Space for Coordinating Mixed-Focus Collaboration."

The 4th ZOIL design principle introduces ZOIL's "device as a camera" metaphor (or ZOIL's "camera concept"). It is based on a discussion of literature about the use of space and territoriality when collaborating at tables or interactive tabletops. This literature concludes that there is a close relation between the availability of space and collaborative coupling styles. People naturally partition their interactions with little to no verbal negotiation into spatial territories. Tabletops should always provide appropriate table space to support this and to avoid negative impacts on collaboration or interference due to overlapping territories.

As a design contribution, ZOIL's 4th design principle transfers these ideas from the physical space on a table to the virtual space in a ZUI: All active devices with a display (e.g., mobile devices, tabletops) can serve as individual "cameras" to provide personal or shared views of the zoomable information landscape. They can be individually controlled to zoom and pan to a desired region and to view it at the desired scale. Collaborators can use ZOIL's "device as a camera" metaphor to access different, same, or overlapping regions of the shared workspace at any time.

This concept supports many different collaboration and coupling styles by enabling a flexible and natural *ad hoc* partitioning of the virtual workspace. Users can create an infinite number of personal, group, or storage territories in the landscape by moving to a certain location in space and scale and starting to collaborate there. Spatial navigation between these territories supports fluid transitions between different coupling styles during mixed-focus collaboration.

Limitations and Future Work

At this stage of research on ZOIL, the assumed benefits of ZOIL's camera concept for coordinating collaboration are only theoretical and are not based on empirical user studies with ZOIL example prototypes. Therefore the contribution of the 4th design principle is limited to the suggestion of a novel design for providing virtual space for co-located collaboration in a multi-user and multi-display environment. This suggestion should be further explored in future work. For example, future user studies could examine, if the virtual navigation in a ZUI has enough similarities with the physical navigation in front of a large display or tabletop to recreate a similarly fluid and natural partitioning of the virtual workspace during collaboration (see discussion in section 5.9.3).

7.1.5 P#5: "Provide Post-WIMP InfoVis Tools for Fluid Interaction."

The 5th ZOIL design principle is based on my cognitive account of post-WIMP fluid interaction with InfoVis from section 6.2:

InfoVis UIs can enhance the user experience with particular engaging, compelling, and even absorbing user experiences that turn the analytical sensemaking process into a pleasurable task. Ideally, they achieve a fluid interaction based on directly interacting in a model-world instead of conversing with an UI using an interface language as

intermediary. Such model-world interfaces promote "staying in the flow" and achieve a feeling of directness and direct manipulation by bridging the gulfs of execution and evaluation. Since today's post-WIMP user interfaces employ a stronger form of direct manipulation than the traditional drag-and-drop of desktop computing with a mouse, they are a particularly promising approach for achieving fluid interaction with InfoVis. They use modalities such as body movement, gestures, multi-touch interaction, or tangible objects to make use of our full range of innate or learned cognitive abilities. This cognitive account of fluid interaction is the first scientific contribution of chapter 6 and is based on my theoretical discussion of different cognitive models in HCI there. It was part of a previous publication in (*Elmqvist, Vande Moere, Jetter et al. 2011*).

This theoretical discussion was inspired by my experiences with designing the post-WIMP InfoVis tabletop system *Facet-Streams* and the results of its evaluation which are the second scientific contribution of chapter 6. This contribution was previously published in (*Jetter, Gerken, Zöllner et al. 2011*). *Facet-Streams* combines techniques of information visualization with tangible and multi-touch interaction to materialize collaborative faceted search on a tabletop. The user studies revealed that users were able to quickly learn and apply the visual-tangible metaphor for Boolean logic and faceted search. *Facet-Streams* proved to be equally effective as established designs for faceted navigation on the Web, although it introduced novel and unfamiliar hybrid interaction techniques and visual metaphors. Users succeeded in translating natural language instructions in complex Boolean queries. This demonstrated how abstract data and logic can be materialized using hybrid visual-tangible user interfaces to make them easier to understand and manipulate for the users. As a result, the users perceived the system's design as appealing, innovative, exciting, likeable, and useful. Overall users perceived using *Facet-Streams* as a fun experience.

Limitations and Future Work

Future work should address the observed usability problems of *Facet-Streams* during browsing results and a closer integration of browsing and query formulation. One possible direction for future designs is to integrate further interactive surfaces, e.g., shared vertical displays, into the system (see Figure 35, p. 47).

7.1.6 P#6: "Support Multi-User Collaboration with Visual-Tangible Externalizations."

The 6th ZOIL design principle is also based on the observations made during the user study of *Facet-Streams* that have been previously published in (*Jetter, Gerken, Zöllner et al. 2011*). In *Facet-Streams*, the shared workspace on the tabletop with visual-tangible networks of personal and collective search criteria served as a successful visual-tangible externalization of collaboration. It provided a map of the search process that captured its (chrono-)logical development. Users used it as an indicator of their progress and to store, revisit, and reuse intermediate results. This led to an increased awareness and to close collaboration that showed qualities of a fluid interaction with first steps towards a collaborative "flow" experience. Furthermore, users were able to attribute tokens and their criteria to other collaborators based on the topology and spatial distribution of the

networks. Therefore the visual-tangible externalization enabled a great variety of search strategies and collaboration styles and supported seamless transitions between tightly-coupled collaboration and loosely-coupled parallel work. In conclusion, this design serves as an example for a successful use of tangible user interface elements for collaborative information visualization and can inform future designs of collaborative interactive spaces.

Limitations and Future Work

Users often had to move tokens out of the tabletop's center, since they occluded results or felt too distracting. In future, occlusion by tangible elements should be minimized, for example by using transparent fiducial markers with IR-absorbing ink or using greater tabletops to increase the visible region.

Furthermore, future work could focus specifically on studying the benefit of tangible UI elements in *Facet-Streams*. By comparing two variants of Facet-Streams (one with tangible tokens and one using only virtual tokens that are moved by touch) the actual effect of tangible vs. non-tangible interaction could be isolated and analyzed.

7.2 The ZOIL Software Framework

Another component of the ZOIL paradigm is the ZOIL software framework that is introduced and evaluated in chapter 3. The framework provides all the necessary core functionality for implementing ZOIL-based OOUIs for a post-WIMP multi-user and multi-device interactive space. It provides high-level functionality in the three areas of *presentation & interaction*, *communication*, and *persistence & synchronization*. These key aspects of ZOIL-based interactive spaces can be realized by using the existing software components from the ZOIL framework without the need for extensive knowledge about the details of the underlying lower level libraries or APIs of the operating system.

Key features of the framework are the easy implementation of semantic zooming into objects that are contained in a shared zoomable visual workspace, real-time synchronization of the workspace across device boundaries, the declarative definition of the objects' visual appearance and interactive behaviors with XAML, and communication and view coupling between different devices using OSC.

The ZOIL software framework's contribution is primarily technological. Similar to other frameworks in HCI, e.g., Jeffrey Heer's prefuse, it is aimed at facilitating implementation and prototyping for researchers and practitioners. It is therefore shared for reuse in academia and industry as open-source under the BSD License. The practical value of the framework is documented by its acceptance among other researchers. To this day, two PhD theses (not including this thesis), five Master's theses, two Bachelor's theses, two journal publications, and nine peer-reviewed conference papers with two honorable mention awards used the ZOIL software framework for prototyping (see Figure 62, p.100). During API usability evaluation, participants rated the ceiling ("how much can be done using the tool?") of the ZOIL software framework slightly above neutral, hinting at

limitations and room for improvement that are discussed below. Participants rated ZOIL's threshold ("how easy it is to learn how to use the tool?") slightly below neutral. However, analysis of the development effort revealed that ZOIL's threshold can still be considered low enough that a 2 person team of C# and WPF novices is able to create a multi-device post-WIMP ZOIL UI in a fulltime project of 2-3 weeks. In conclusion, the ZOIL software framework can be considered an appropriate tool for the prototyping and implementation of post-WIMP interactive spaces and thus has been part of previous publications (*Jetter, Gerken, Zöllner et al. 2010b*; *Jetter, Zöllner, and Reiterer 2011*; *Jetter, Zöllner, Gerken et al. 2012*) .

A scientific rather than a purely technological contribution are the findings of the API usability study about monolithic vs. polylithic architectures and the participants' preference for concrete mappings. They hint at a more general theme in post-WIMP implementation. Similar to the user of a post-WIMP UI who perceives an object as a single physical entity (e.g., a virtual or tangible object on a tabletop with size, mass, friction), post-WIMP programmers would like to create such objects and define their behaviors and properties without creating many abstract artificial software objects that have no counterparts in the real world, for example, when using patterns like *Model-View-ViewModel* or *Views-Objects-Governors-Instruments*. While participants in the study accepted the necessity for a separation of view and model, they expressed that they still would strongly prefer more natural mappings between code objects and real-world objects and more declarative approaches for defining properties and behaviors.

Limitations and Future Work

As discussed in section 7.1.1, a limitation of the API usability evaluation is the absence of professional designers and developers. In future work, a comparable study with a larger scale project and more experienced designers and developers as participants could reveal how the ZOIL software framework performs in a professional environment.

Furthermore, the study revealed limitations of the ZOIL software framework that decreased its perceived ceiling and increased its perceived threshold. Since the development and extension of the ZOIL software framework is an ongoing activity beyond this PhD project, future versions of ZOIL will attempt to overcome these limitations. Based on my assessment, the first points to address are the rendering performance when displaying large amounts of information objects (see section 4.4) and better hosting of Web content and ActiveX components in the ZUI including scaling and rotation (see section 3.5.5). Of lesser priority are the reported lack of drag and drop support (see section 3.5.5) and the problems with floating point precision at large scale factors (see section 4.4) since they can also be addressed on application level.

For sustaining the value of the ZOIL framework in future device ecologies, I also suggest to consider porting the ZOIL software framework to Microsoft's new *Windows Runtime* ("WinRT") that is part of Windows 8. WinRT natively supports the x86 and ARM CPU architectures and is intended to replace Microsoft's .NET platform on the coming generation of portable Windows devices including tablets and smart phones.

7.3 Conclusion

From its outset, my PhD project and PhD thesis about the ZOIL paradigm touched a broad range of different fields related to human-computer interaction including ubiquitous computing, information visualization, computer-supported cooperative work, cognitive science, personal information management, and software engineering. Therefore it has been a challenge to present my contributions and results in a concise, structured, and comprehensible manner.

As mentioned in the introduction, this thesis about the ZOIL paradigm is my attempt to contribute to the improvement of present-day ubiquitous computing by not only writing for an academic audience, but also providing practitioners with hands-on designs, examples, guidelines, and tools. I hope that this thesis succeeds in striking a good balance between the theory, design, technology, and practice of post-WIMP interactive spaces.

Weiser formulated his vision of a seamless interaction across device boundaries by arguing that *"real power (...) comes not from any of these devices – it emerges from the interaction of all of them"* (*Weiser 1991*). I hope that the ZOIL paradigm contributes to achieving Weiser's vision of a powerful, yet invisible and calm, ubicomp that finally makes computers fit the human environment, so that we are freed to use them without thinking and to focus beyond them on new goals.

8 References

Aarts, Emile H. L (2003), 'Ambient Intelligence: First European Symposium. Proceedings.', in Emile H. L. Aarts (ed.), *EUSAI 2003* (LNCS 2875; Veldhoven, the Netherlands: Springer).

Ahlberg, Christopher and Shneiderman, Ben (1994), 'Visual information seeking using the FilmFinder', *Conference companion on Human factors in computing systems* (Boston, Massachusetts, United States: ACM), 433-34.

Amershi, Saleema and Morris, Meredith Ringel (2008), 'CoSearch: a system for co-located collaborative web search', *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems* (Florence, Italy: ACM), 1647-56.

Andrews, Christopher, Endert, Alex, and North, Chris (2010), 'Space to think: large high-resolution displays for sensemaking', *Proceedings of the 28th international conference on Human factors in computing systems* (Atlanta, Georgia, USA: ACM), 55-64.

Baddeley, Alan D. (1986), *Working Memory* (Oxford: Oxford University Press).

Bakker, Niels H., Werkhoven, Peter J., and Passenier, Peter O. (1999), 'The Effects of Proprioceptive and Visual Feedback on Geographical Orientation in Virtual Environments', *Presence: Teleoper. Virtual Environ.*, 8 (1), 36-53.

Balakrishnan, Ravin and Baudisch, Patrick (2009), 'Introduction to this Special Issue on Ubiquitous Multi-Display Environments', *Human–Computer Interaction,* 24 (1-2), 1-8.

Beaudouin-Lafon, Michel (2000), 'Instrumental interaction: an interaction model for designing post-WIMP user interfaces', *Proc. CHI '00* (2: ACM), 446-53.

Beck, Astrid, Janssen, Christian, Weisbecker, Anette, and Ziegler, Jürgen (1994), 'Integrating object-oriented analysis and graphical user interface design.', in R. N. Taylor and J. Coutaz (eds.), *ICSE-WS 1994 and SE-HCI 1994* (LNCS, vol. 896; Sorrento, Italy: Springer), 127-40.

Bederson, Benjamin B. (2001), 'PhotoMesa: a zoomable image browser using quantum treemaps and bubblemaps', *Proceedings of the 14th annual ACM symposium on User interface software and technology* (Orlando, Florida: ACM), 71-80.

Bederson, Benjamin B. and Boltman, Angela (1999), 'Does Animation Help Users Build Mental Maps of Spatial Information?', *Proceedings of the 1999 IEEE Symposium on Information Visualization* (IEEE Computer Society), 28.

Bederson, Benjamin B., Grosjean, Jesse, and Meyer, Jonathan (2004), 'Toolkit Design for Interactive Structured Graphics', *IEEE Transactions on Software Engineering,* 30, 535-46.

Bederson, Benjamin B., Clamage, Aaron, Czerwinski, Mary P., and Robertson, George G. (2004), 'DateLens: A fisheye calendar interface for PDAs', *ACM Trans.Comput.-Hum.Interact.,* 11, 90-119.

Bederson, Benjamin B., Hollan, James D., Perlin, Ken, Meyer, Jonathan, Bacon, David, and Furnas, George W. (1996), 'Pad++: A Zoomable Graphical Sketchpad For Exploring Alternate Interface Physics', *Journal of Visual Languages and Computing,* 7, 3-32.

Bell, Genevieve and Dourish, Paul (2006), 'Yesterday's tomorrows: notes on ubiquitous computing's dominant vision', *Personal and Ubiquitous Computing,* 11, 133-43.

Benyon, David and Mival, Oli (2012), 'Blended Spaces for Collaborative Creativity', *Designing Collaborative Interactive Spaces (an AVI 2012 workshop)* (Capri Italy: Human-Computer Interaction Group, University of Konstanz).

Berry, Richard E. and Reeves, Cliff J. (1992), 'The evolution of the Common User Access workplace model', *IBM Syst. J.*, 31, 414-28.

Biehl, Jacob T. and Bailey, Brian P. (2004), 'ARIS: an interface for application relocation in an interactive space', *Proceedings of Graphics Interface 2004* (London, Ontario, Canada: Canadian Human-Computer Communications Society), 107-16.

Biehl, Jacob T., Baker, William T., Bailey, Brian P., Tan, Desney S., Inkpen, Kori M., and Czerwinski, Mary (2008), 'Impromptu: a new interaction framework for supporting collaboration in multiple display environments and its field evaluation for co-located software development', *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems* (Florence, Italy: ACM), 939-48.

Bier, Eric A., Stone, Maureen C., Pier, Ken, Buxton, William, and DeRose, Tony D. (1993), 'Toolglass and magic lenses: the see-through interface', *Proceedings of the 20th annual conference on Computer graphics and interactive techniques* (Anaheim, CA: ACM), 73-80.

Blackwell, Alan F. and Green, Thomas (2003), 'Notational Systems – The Cognitive Dimensions of Notations Framework', in J. M. Carroll (ed.), *HCI Models, Theories, and Frameworks: Toward a Multidisciplinary Science* (San Francisco: Morgan Kaufmann), 103-34.

Blackwell, Alan F., Stringer, Mark, Toye, Eleanor F., and Rode, Jennifer A. (2004), 'Tangible interface for collaborative information retrieval', *CHI '04 extended abstracts on Human factors in computing systems* (Vienna, Austria: ACM), 1473-76.

Blandford, Ann and Attfield, Simon (2010), 'Interacting with Information', in John M Carroll (ed.), *Synthesis Lectures on Human-Centered Informatics* (Morgan & Claypool).

Bolt, Richard A. (1984), *The human interface.* (Belmont, CA, USA: Lifetime Learning Publications).

Borgman, Christine L. (1996), 'Why are online catalogs *still* hard to use?', *J. Am. Soc. Inf. Sci.*, 47 (7), 493-503.

Budzinski, Jochen (2012), 'Hybrid Brainsketching - Interaction Concepts for Sketching Activities Based on Digital Pen & Paper and Interactive Visualizations', Master's Thesis (University of Konstanz).

Büring, Thorsten , Gerken, Jens, and Reiterer, Harald (2006), 'Usability of overview-supported zooming on small screens with regard to individual differences in spatial ability', *Proceedings of the working conference on Advanced visual interfaces* (Venezia, Italy: ACM), 233-40.

Butz, Andreas and Krüger, Antonio (2006), 'Applying the Peephole Metaphor in a Mixed-Reality Room', *IEEE Comput. Graph. Appl.*, 26 (1), 56-63.

Butz, Andreas, Höllerer, Tobias, Beshers, Clifford, Feiner, Steven K., and MacIntyre, Blair (1999), 'An Experimental Hybrid User Interface for Collaboration', *Columbia University Computer Science Technical Reports* (CUCS-005-99; New York: Columbia University, Department of Computer Science).

Card, Stuart K. (2008), 'Information visualization', in Julie A Jacko and Andrew Sears (eds.), *The Human Computer Interaction Handbook* (2nd edn.; Mahwah, NJ, USA: Lawrence Erlbaum Associates, Inc), 509-43.

Card, Stuart K., Mackinlay, Jock D., and Shneiderman, Ben (1999), 'Information visualization', in K. Card Stuart, D. Mackinlay Jock, and Shneiderman Ben (eds.), *Readings in information visualization* (Morgan Kaufmann Publishers Inc.), 1-34.

Cockburn, Andy, Karlson, Amy K., and Bederson, Benjamin B. (2008), 'A review of overview+detail, zooming, and focus+context interfaces', *ACM Computing Surveys (CSUR),* 41, 1-31.

Collins, Dave (1995), *Designing object-oriented user interfaces* (Redwood City, CA: Benjamin Cummings).

Constantine, Larry (2002), 'The Emperor Has No Clothes : Naked Objects Meet the Interface', <*http://www.foruse.com/articles/nakedobjects.htm*>, accessed Aug 23, 2011.

Constantine, Larry and Lockwood, Lucy (1999), *Software for use : a practical guide to the models and methods of usage-centered design* (Reading, Mass.: Addison Wesley).

Cook, Diane and Das, Sajal (2004), *Smart Environments: Technology, Protocols and Applications* (Wiley).

Csikszentmihalyi, Mihaly (1990), *Flow : the psychology of optimal experience* (1st edn.; New York: Harper & Row) xii, 303 p.

Dahl, Ole-Johan and Nygaard, Kristen (1966), 'SIMULA: an ALGOL-based simulation language', *Commun. ACM,* 9 (9), 671-78.

Darken, Rudolph P. and Peterson, Barry (2001), 'Spatial orientation, Wayfinding, and Representation', in K Stanney (ed.), *Handbook of Virtual Environment Technology* (New Jersey: Lawrence Erlbaum Assoc.), 493-518.

De Ruiter, J.-P. (1998), 'Gesture and speech production', PhD Thesis (Max Planck Institute for Psycholinguistics).

DeLine, Robert and Rowan, Kael (2010), 'Code canvas: zooming towards better development environments', *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 2* (Cape Town, South Africa: ACM), 207-10.

Demarmels, Mischa (2010), 'Enhancing Collaborative Information-Seeking through Surface and Tangible Computing', Master's Thesis (University of Konstanz).

Demarmels, Mischa, Huber, Stephan, and Heilig, Mathias (2010), 'Meet Me in the Library!', *Proc. SIDeR '10* (Umeå, Sweden: Umeå Institute of Design), 61-66.

Detken, Karen, Martinez, Carlos, and Schrader, Andreas (2009), 'The search wall: tangible information searching for children in public libraries', *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction* (Cambridge, United Kingdom: ACM), 289-96.

Dietz, Paul and Leigh, Darren (2001), 'DiamondTouch: a multi-user touch technology', *Proceedings of the 14th annual ACM symposium on User interface software and technology* (Orlando, Florida: ACM), 219-26.

Donelson, William C. (1978), 'Spatial management of information', *SIGGRAPH '78: Proceedings of the 5th annual conference on Computer graphics and interactive techniques* (ACM), 203-09.

Dosi, Giovanni (1988), 'Sources, Procedures, and Microeconomic Effects of Innovation', *Journal of Economic Literature,* 26, 1120-71.

Dourish, Paul (2004), *Where the Action Is: The Foundations of Embodied Interaction* (First MIT Press Paperback Edition edn.: MIT Press) 233.

Dourish, Paul and Bellotti, Victoria (1992), 'Awareness and coordination in shared workspaces', *Proceedings of the 1992 ACM conference on Computer-supported cooperative work* (Toronto, Ontario, Canada: ACM), 107-14.

Drucker, P. F. (1973), *Management: Tasks, Responsibilities and Practices* (New York: Harper & Row).

Ebert, Achim, Deller, Matthias, Steffen, Daniel, and Heintz, Matthias (2009), '"Where Did I Put That?" – Effectiveness of Kinesthetic Memory in Immersive Virtual Environments', in Constantine Stephanidis (ed.), *Universal Access in Human-Computer Interaction. Applications and Services* (Lecture Notes in Computer Science, 5616: Springer Berlin / Heidelberg), 179-88.

Ellis, Clarence A., Gibbs, Simon J., and Rein, Gail (1991), 'Groupware: some issues and experiences', *Commun. ACM,* 34 (1), 39-58.

Elmqvist, Niklas, Vande Moere, Andrew, Jetter, Hans-Christian, Cernea, Daniel, Reiterer, Harald, and Jankun-Kelly, T.J. (2011), 'Fluid Interaction for Information Visualization', *Information Visualization,* 10 (4), 327-40.

Endres, Christoph, Butz, Andreas, and MacWilliams, Asa (2005), 'A survey of software infrastructures and frameworks for ubiquitous computing', *Mob. Inf. Syst.,* 1 (1), 41-80.

Engl, Andreas (2008), 'A Framework for an Infinitely Zoomable Information Landscape', Bachelor's Thesis (University of Konstanz).

Evans, Eric (2004), *Domain-driven design : tackling complexity in the heart of software* (Addison-Wesley).

Fäh, Simon (2011), 'Bimanual Pointing Techniques for Cross-Display Interaction', Master's Thesis (University of Konstanz).

Fitzmaurice, George W. (1993), 'Situated information spaces and spatially aware palmtop computers', *Commun. ACM,* 36 (7), 39-49.

Forlines, Clifton, Wigdor, Daniel, Shen, Chia, and Balakrishnan, Ravin (2007), 'Direct-touch vs. mouse input for tabletop displays', *Proceedings of the SIGCHI conference on Human factors in computing systems* (San Jose, California, USA: ACM), 647-56.

Frank M. Shipman, III, Hsieh, Haowei, Maloor, Preetam, and Moore, J. Michael (2001), 'The visual knowledge builder: a second generation spatial hypertext', *Proceedings of the 12th ACM conference on Hypertext and Hypermedia* (Arhus, Denmark: ACM), 113-22.

Fuks, Hugo, Raposo, Alberto, Gerosa, Marco A., Pimental, Mariano, and Lucena, Carlos J. P. (2008), 'The 3C Collaboration Model', in Ned Cock (ed.), *Encyclopedia of E-collaboration* (Information Science Reference), 637-43.

Furnas, George W. and Bederson, Benjamin B. (1995), 'Space-scale diagrams', *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '95* (New York, New York, USA: ACM Press), 234-41.

Gerken, Jens (2011), 'Longitudinal Research in Human-Computer Interaction', PhD Thesis (University of Konstanz).

Gerken, Jens, Jetter, Hans-Christian, and Schmidt, Toni (2010), 'Can "touch" get annoying?', *Proc. ITS 2010 (Poster Session)* (ACM), 257-58.

Gerken, Jens, Jetter, Hans-Christian, Zöllner, Michael, Mader, Martin, and Reiterer, Harald (2011), 'The concept maps method as a tool to evaluate the usability of APIs', *Proc. CHI '11* (ACM), 3373-82.

Gerken, Jens, Heilig, Mathias, Jetter, Hans-Christian, Rexhausen, Sebastian, Demarmels, Mischa, König, Werner A, and Reiterer, Harald (2009), 'Lessons Learned from the Design and Evaluation of Visual Information Seeking Systems', *International Journal on Digital Libraries,* 10, 49-66.

Geyer, Florian and Reiterer, Harald (2010), 'A Cross-Device Spatial Workspace Supporting Artifact-Mediated Collaboration in Interaction Design', *CHI EA '10*, 3787-92.

Geyer, Florian, Budzinski, Jochen, and Reiterer, Harald (2012), 'IdeaVis: A Hybrid Workspace and Interactive Visualization for Paper-based Collaborative Sketching Sessions', *Proceedings of the 7th Nordic Conference on Human-Computer Interaction (NordiCHI '12)* (Copenhagen, Denmark: ACM Press), 331-40.

Geyer, Florian, Jetter, Hans-Christian, Pfeil, Ulrike, and Reiterer, Harald (2010), 'Collaborative Sketching with Distributed Displays and Multimodal Interfaces', *Proc. ITS 2010 (Poster Session)* (Saarbrücken, Germany: ACM), 259-60.

Geyer, Florian, Pfeil, Ulrike, Höchtl, Anita, Budzinski, Jochen, and Reiterer, Harald (2011), 'Designing Reality-Based Interfaces for Creative Group Work', *C&C'11: Proceedings of the 8th ACM Conference on Creativity and Cognition* (Atlanta, GA, USA: ACM), 165-74.

Geyer, Florian, Pfeil, Ulrike, Budzinski, Jochen, Höchtl, Anita, and Reiterer, Harald (2011), 'AffinityTable - a hybrid surface for supporting affinity diagramming', *Proceedings of the 13th IFIP TC 13 international conference on Human-computer interaction - Volume Part III* (Lisbon, Portugal: Springer-Verlag), 477-84.

Gibbs, Raymond W (2006), *Embodiment and Cognitive Science* (Psychology: Cambridge University Press) 337.

Gjerlufsen, Tony, Klokmose, Clemens Nylandsted, Eagan, James, Pillias, Clément, and Beaudouin-Lafon, Michel (2011), 'Shared substance: developing flexible multi-surface applications', *Proceedings of the 2011 annual conference on Human factors in computing systems* (Vancouver, BC, Canada: ACM), 3383-92.

Goldberg, Adele (1990), 'Information models, views, and controllers', *Dr. Dobb's J.,* 15 (7), 54-61.

Goldberg, Adele and Robson, David (1983), *Smalltalk-80: the language and its implementation* (Addison-Wesley Longman Publishing Co., Inc.) 742.

Goldstine, Herman H (1972), *The Computer from Pascal to von Neumann* (Princeton University Press).

Greenberg, Saul (1989), 'The 1988 Conference on Computer-Supported Cooperative Work: Trip Report', *ACM SIGCHI Bulletin,* Vol. 20 of 5, 49-55.

Greenberg, Saul, Marquardt, Nicolai, and Ballendat, T (2011), 'Proxemic interactions: the new ubicomp?', *interactions*, 42-50.

Guiard, Yves and Beaudouin-Lafon, Michel (2004), 'Target Acquisition in Multiscale Electronic Worlds', *International Journal of Human-Computer Studies,* 61 (6), 875-905.

Guo, Huo, Zhang, Vivian, and Wu, Jing 'The Effect of Zooming Speed in a Zoomable User Interface', <*http://otal.umd.edu/SHORE2000/zoom/index.html>*, accessed Jan 5th, 2011.

Gutwin, Carl and Greenberg, Saul (1999), 'The effects of workspace awareness support on the usability of real-time distributed groupware', *ACM Trans. Comput.-Hum. Interact.,* 6 (3), 243-81.

Haller, Michael, Leitner, Jakob, Seifried, Thomas, Wallace, James R., Scott, Stacey D., Richter, Christoph, Brandl, Peter, Gokcezade, Adam, and Hunter, Seth (2010), 'The NiCE Discussion Room: Integrating Paper and Digital Media to Support Co-Located Group Meetings', *Proceedings of the 28th international conference on Human factors in computing systems* (Atlanta, Georgia, USA: ACM), 609-18.

Hansaki, Tomoyuki, Shizuki, Buntarou, Misue, Kazuo, and Tanaka, Jiro (2006), 'FindFlow: visual interface for information search based on intermediate results', *Proceedings of the 2006 Asia-Pacific Symposium on Information Visualisation - Volume 60* (Tokyo, Japan: Australian Computer Society, Inc.), 147-52.

Hansen, Wilfred J. (1971), 'User engineering principles for interactive systems', *Proceedings of the November 16-18, 1971, fall joint computer conference* (Las Vegas, Nevada: ACM), 523-32.

Harper, Richard, Rodden, Tom, Rogers, Yvonne, and Sellen, Abigail (2008), *Being human: Human-computer interaction in the year 2020* (Cambridge, England: Microsoft Research Ltd).

Hartmann, Björn, Morris, Meredith Ringel, Benko, Hrvoje, and Wilson, Andrew D. (2009), 'Augmenting interactive tables with mice & keyboards', *Proceedings of the 22nd annual ACM symposium on User interface software and technology* (Victoria, BC, Canada: ACM), 149-52.

Haywood, Dan (2009), *Domain-Driven Design Using Naked Objects* (Raleigh, NC: Pragmatic Bookshelf) 375.

Hearst, Marti (2009), *Search user interfaces* (Cambridge ; New York: Cambridge University Press) xviii, 385 p., 12 p. of plates.

Hearst, Marti A. (2011), ''Natural' search user interfaces', *Commun. ACM,* 54 (11), 60-67.

Heer, Jeffrey, Card, Stuart K., and Landay, James A. (2005), 'prefuse: a toolkit for interactive information visualization', *Proceedings of the SIGCHI conference on Human factors in computing systems* (Portland, Oregon, USA: ACM), 421-30.

Heilig, Mathias (2012), 'Exploring Reality-Based User Interfaces for Collaborative Information Seeking', PhD Thesis (University of Konstanz).

Heilig, Mathias, Demarmels, Mischa, Rexhausen, Sebastian, Huber, Stephan, and Runge, Oliver (2009), 'Search, Explore and Navigate - Designing a Next Generation Knowledge Media Workbench', in I.H.C. Wouters, et al. (eds.), *Fifth Student Interaction Design Research Conference (SIDeR 09) - Flirting with the future* (Eindhoven University of Technology, the Netherlands), 40-43.

Heilig, Mathias, Demarmels, Mischa, Allmendinger, Katrin, Gerken, Jens, and Reiterer, Harald (2010), 'Fördern realitätsbasierte UIs kollaborative Rechercheaktivitäten?', *Proceedings of Mensch & Computer 2010 - Interaktive Kulturen* (Duisburg: Oldenbourg), 311-20.

Heilig, Mathias, Huber, Stephan, Gerken, Jens, Demarmels, Mischa, Allmendinger, Katrin, and Reiterer, Harald (2011), 'Hidden details of negotiation: the mechanics of reality-based collaboration in information seeking', *Proceedings of the 13th IFIP TC 13 international conference on Human-computer interaction - Volume Part II* (Lisbon, Portugal: Springer-Verlag), 622-39.

Hofmeester, Kay and Wixon, Dennis (2010), 'Using metaphors to create a natural user interface for microsoft surface', *Proceedings of the 28th of the international conference extended abstracts on Human factors in computing systems* (Atlanta, Georgia, USA: ACM), 4629-44.

Hollan, James, Hutchins, Edwin, and Kirsh, David (2000), 'Distributed cognition: toward a new foundation for human-computer interaction research', *ACM Trans. Comput.-Hum. Interact.,* 7 (2), 174-96.

Hornbaek, Kasper, Bederson, Benjamin B., and Plaisant, Catherine (2002), 'Navigation patterns and usability of zoomable user interfaces with and without an overview', *ACM Trans. Comput.-Hum. Interact.,* 9 (4), 362-89.

Hornecker, Eva and Buur, Jacob (2006), 'Getting a grip on tangible interaction: a framework on physical space and social interaction', *Proceedings of the SIGCHI conference on Human Factors in computing systems* (Montreal, Quebec, Canada: ACM), 437-46.

Hutchins, E.L. (2001), 'Distributed cognition', in N. J. Smelser and P. B. Baltes (eds.), *The International Encyclopedia of the Social and Behavioral Sciences* (Elsevier), pp. 2068-72.

Hutchins, E.L., Hollan, J.D., and Norman, D.A. (1985), 'Direct manipulation interfaces', *Human–Computer Interaction,* 1, 311-38.

Huuskonen, Pertti (2007), 'Run to the hills! ubiquitous computing meltdown', *Proceeding of the 2007 conference on Advances in Ambient Intelligence* (IOS Press), 157-72.

Imaz, Manuel and Benyon, David (2007), *Designing with Blends: Conceptual Foundations of Human-Computer Interaction and Software Engineering* (MIT Press).

Isenberg, Petra and Fisher, Danyel (2011), 'Cambiera: collaborative tabletop visual analytics', *Proceedings of the ACM 2011 conference on Computer supported cooperative work* (Hangzhou, China: ACM), 581-82.

Izadi, Shahram, Brignull, Harry, Rodden, Tom, Rogers, Yvonne, and Underwood, Mia (2003), 'Dynamo: a public interactive surface supporting the cooperative sharing and exchange of media', *Proceedings of the 16th annual ACM symposium on User interface software and technology* (Vancouver, Canada: ACM), 159-68.

Jacob, Robert J. K., Girouard, Audrey, Hirshfield, Leanne M., Horn, Michael S., Shaer, Orit, Solovey, Erin Treacy, and Zigelbaum, Jamie (2007), 'Reality-based interaction: unifying the new generation of interaction styles', *CHI '07 extended abstracts on Human factors in computing systems* (San Jose, CA, USA: ACM), 2465-70.

Jacob, Robert J. K., Girouard, Audrey, Hirshfield, L.M., Horn, M.S., Shaer, O., Solovey, E.T., and Zigelbaum, Jamie (2008), 'Reality-based interaction: a framework for post-WIMP interfaces', *Proc. CHI '08* (ACM), 201-10.

Jenabi, Mahsa (2011), 'PrIME: Primitive Interaction Tasks for Multi-Display Environments', PhD Thesis (University of Konstanz).

Jetter, Hans-Christian and Gerken, Jens (2006), 'A Simplified Model of User Experience for Practical Application', *The 2nd COST294-MAUSE International Open Workshop User eXperience - Towards a unified view (a NordiCHI 2006 workshop).* (Oslo, Norway: COST294-MAUSE), 106-11.

Jetter, Hans-Christian, König, Werner A., and Reiterer, Harald (2009), 'Understanding and Designing Surface Computing with ZOIL and Squidy', *Multitouch and Surface Computing (a CHI 2009 Workshop)* <http://hci.uni-konstanz.de/downloads/SurfaceComputing_Workshop_CHI09.pdf>, accessed Aug 23, 2011.

Jetter, Hans-Christian, Zöllner, Michael, and Reiterer, Harald (2011), 'Gestaltung und Programmierung von interaktiven Räumen mit dem ZOIL-Paradigma', in Rainer Groh and Martin Zavesky (eds.), *Wieder mehr sehen! Aktuelle Einblicke in die Technische Visualistik* (Dresden, Germany: TUDpress), 105-15.

Jetter, Hans-Christian, Engl, Andreas, Schubert, Soeren, and Reiterer, Harald (2008), 'Zooming not zapping: Demonstrating the ZOIL User Interface Paradigm for ITV Applications', *Adjunct Proceedings of EuroITV 2008* (Salzburg, Austria: Springer), 239-40.

Jetter, Hans-Christian, König, Werner A., Gerken, Jens, and Reiterer, Harald (2008), 'ZOIL - A Cross-Platform User Interface Paradigm for Personal Information Management', *The Disappearing Desktop: Personal Information Management 2008 (a CHI 2008 Workshop)* <http://nbn-resolving.de/urn:nbn:de:bsz:352-opus-75367 >, accessed Aug 23, 2011.

Jetter, Hans-Christian, Gerken, Jens, Zöllner, Michael, and Reiterer, Harald (2010a), 'Be-Greifbare Interaktion für die kollaborative Suche', *Interaktive Kulturen Workshop-Band. Proceedings der Workshops der Mensch & Computer 2010.* (Logos Verlag), 145-50.

--- (2010b), 'Model-based design and implementation of interactive spaces for information interaction', *Proc. Human-Centred Software Engineering (HCSE '10)* (Reykjavik, Iceland: Springer), 22-37.

Jetter, Hans-Christian, Zöllner, Michael, Gerken, Jens, and Reiterer, Harald (2012), 'Design and Implementation of Post-WIMP Distributed User Interfaces with ZOIL', *International Journal of Human-Computer Interaction,* 28 (11), 737-47.

Jetter, Hans-Christian, Gerken, Jens, König, Werner A., Grün, Christian, and Reiterer, Harald (2005), 'HyperGrid - Accessing Complex Information Spaces', *BCS-HCI UK 2005* (Edinburgh, UK: Springer), 349-64.

Jetter, Hans-Christian, Gerken, Jens, Zöllner, Michael, Reiterer, Harald, and Milic-Frayling, Natasa (2011), 'Materializing the query with facet-streams: a hybrid surface for collaborative search on tabletops', *Proc. CHI '11 (Honorable Mention Paper Award)* (ACM), 3013-22.

Jetter, Hans-Christian, Leifert, Svenja, Gerken, Jens, Schubert, Sören, and Reiterer, Harald (2012), 'Does (Multi-)Touch Aid Users' Spatial Memory and Navigation in 'Panning' and in 'Zooming & Panning' UIs?', *Proceedings of the International Working Conference on Advanced Visual Interfaces* (Capri Island, Italy: ACM Press), 83-90.

Johansen, Robert (1988), *GroupWare: Computer Support for Business Teams* (The Free Press) 256.

Johanson, Brad, Fox, Armando, and Winograd, T. (2002), 'The Interactive Workspaces project: experiences with ubiquitous computing rooms', *IEEE Pervasive Computing,* 1, 67-74.

Johanson, Brad, Hutchins, Greg, Winograd, Terry, and Stone, Maureen (2002), 'PointRight: experience with flexible input redirection in interactive workspaces', *Proceedings of the 15th annual ACM symposium on User interface software and technology* (Paris, France: ACM), 227-34.

Jones, William and Teevan, Jaime (2007), *Personal Information Management* (Seattle, WA: University of Washington Press).

Jordà, Sergi, Geiger, Günter, Alonso, Marcos, and Kaltenbrunner, Martin (2007), 'The reacTable: exploring the synergy between live music performance and tabletop tangible interfaces', *Proceedings of the 1st international conference on Tangible and embedded interaction* (Baton Rouge, Louisiana: ACM), 139-46.

Ju, Wendy and Leifer, Larry (2008), 'The Design of Implicit Interactions: Making Interactive Systems Less Obnoxious', *Design Issues,* 24 (3), 72-84.

Jul, Susanne and Furnas, George W. (1998), 'Critical zones in desert fog: aids to multiscale navigation', *Proceedings of the 11th annual ACM symposium on User interface software and technology* (San Francisco, California, United States: ACM), 97-106.

Kay, Alan and Goldberg, Adele (1977), 'Personal Dynamic Media', *IEEE Computer,* 10, 31-41.

Keim, Daniel A. (2002), 'Information Visualization and Visual Data Mining', *IEEE Transactions on Visualization and Computer Graphics,* 8 (1), 1-8.

Kelso, J. A. Scott (1995), *Dynamic Patterns : The Self-Organization of Brain and Behavior* (Cambridge, MA, USA: MIT Press).

Kidd, Alison (1994), 'The marks are on the knowledge worker', *Proceedings of the SIGCHI conference on Human factors in computing systems: celebrating interdependence* (Boston, Massachusetts, United States: ACM), 186-91.

Kim, KyungTae, Javed, Waqas, Williams, Cary, Elmqvist, Niklas, and Irani, Pourang (2010), 'Hugin: a framework for awareness and coordination in mixed-presence collaborative information visualization', *ACM International Conference on Interactive Tabletops and Surfaces* (Saarbrücken, Germany: ACM), 231-40.

Kirk, David, Sellen, Abigail, Taylor, Stuart, Villar, Nicolas, and Izadi, Shahram (2009), 'Putting the Physical into the Digital : Issues in Designing Hybrid Interactive Surfaces', *Proc. BCS HCI 2009* (Cambridge, UK: British Computer Society), 35-44.

Kirsh, David (1995), 'The intelligent use of space', *Artif. Intell.*, 73 (1-2), 31-68.

Klemmer, S.R., Hartmann, Björn, and Takayama, Leila (2006), 'How bodies matter: five themes for interaction design', *Proceedings of the 6th conference on Designing Interactive systems* (ACM), 140-49.

Klokmose, Clemens Nylandsted and Beaudouin-Lafon, Michel (2009), 'VIGO: instrumental interaction in multi-surface environments', *Proceedings of the 27th international conference on Human factors in computing systems* (Boston, MA, USA: ACM), 869-78.

König, Werner, Rädle, Roman, and Reiterer, Harald (2010), 'Interactive design of multimodal user interfaces', *Journal on Multimodal User Interfaces,* 3 (3), 197-213.

König, Werner A (2006), 'Referenzmodell und Machbarkeitsstudie für ein neues Zoomable User Interface Paradigma', Master's Thesis (University of Konstanz).

Kuhn, Thomas S. (1962), *The structure of scientific revolutions* (Chicago: University of Chicago Press) xv, 172 p.

Lee, Bongshin, Isenberg, Petra, Riche, Nathalie Henry, and Carpendale, Sheelagh (2012), 'Beyond Mouse and Keyboard: Expanding Design Considerations for Information Visualization Interactions', *IEEE Transactions on Visualization and Computer Graphics,* 18, 2689-98.

Lehikoinen, Juha, Aaltonen, Antti, Huuskonen, Pertti, and Salminen, Ilkka (2007), *Personal Content Experience: Managing Digital Life in the Mobile Age* (Chichester, UK: Wiley).

Leifert, Svenja (2011), 'The influence of grids on spatial and content memory', *Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems* (Vancouver, BC, Canada: ACM), 941-46.

Leiner, Barry M., Cerf, Vinton G., Clark, David D., Kahn, Robert E., Kleinrock, Leonard, Lynch, Daniel C., Postel, Jon, Roberts, Lawrence G., and Wolff, Stephen S. (1997), 'The past and future history of the Internet', *Communications of the ACM,* 40, 102-08.

Lin, James, Newman, Mark W., Hong, Jason I., and Landay, James A. (2000), 'DENIM: finding a tighter fit between tools and practice for Web site design', *Proceedings of the SIGCHI conference on Human factors in computing systems* (The Hague, The Netherlands: ACM), 510-17.

MacKenzie, I. Scott and Buxton, William (1992), 'Extending Fitts' law to two-dimensional tasks', *Proceedings of the SIGCHI conference on Human factors in computing systems* (Monterey, California, United States: ACM), 219-26.

Malacria, Sylvain, Lecolinet, Eric, and Guiard, Yves (2010), 'Clutch-free panning and integrated pan-zoom control on touch-sensitive surfaces: the cyclostar approach', *Proceedings of the 28th international conference on Human factors in computing systems* (Atlanta, Georgia, USA: ACM), 2615-24.

Malone, Thomas W. (1982), 'Heuristics for designing enjoyable user interfaces: Lessons from computer games', *Proceedings of the 1982 conference on Human factors in computing systems* (Gaithersburg, Maryland, United States: ACM), 63-68.

Mandel, Theo (1994), *The GUI-OOUI War, Windows vs. OS/2: the designer's guide to human-computer interfaces* (New York, NY, USA: Van Nostrand Reinhold).

Memmel, Thomas (2009), 'User Interface Specification for Interactive Software Systems : Process-, Method- and Tool-Support for Interdisciplinary and Collaborative Requirements Modelling and Prototyping-Driven User Interface Specification', PhD Thesis (University of Konstanz).

Meyer, Shawna, Cohen, Oryx, and Nilsen, Erik (1994), 'Device comparisons for goal-directed drawing tasks', *Conference companion on Human factors in computing systems* (Boston, Massachusetts, United States: ACM), 251-52.

Micire, Mark, Schedlbauer, Martin, and Yanco, Holly (2007), 'Horizontal Selection: An Evaluation of a Digital Tabletop Input Device', *Proceedings of the 13th Americas Conference on Information Systems* (Keystone, CO, USA).

Miller, George A. (1968), 'Psychology and Information', *American Documentation,* 19 (3), 286-89.

Moran, Thomas P. and Zhai, Shumin (2007), 'Beyond the Desktop Metaphor in Seven Dimensions', in Victor Kaptelinin and Mary Czerwinski (eds.), *Beyond the Desktop Metaphor : Desiging Integrated Digital Work Environments* (Cambridge, MA, USA: MIT Press), 335-54.

Morris, Meredith Ringel, Paepcke, Andreas, and Winograd, Terry (2006), 'TeamSearch: Comparing Techniques for Co-Present Collaborative Search of Digital Media', *Proceedings of the First IEEE International Workshop on Horizontal Interactive Human-Computer Systems* (IEEE Computer Society), 97-104.

Morris, Meredith Ringel, Lombardo, Jarrod, and Wigdor, Daniel (2010), 'WeSearch: supporting collaborative search and sensemaking on a tabletop display', *Proceedings of the 2010 ACM conference on Computer supported cooperative work* (Savannah, Georgia, USA: ACM), 401-10.

Morris, Meredith Ringel, Fisher, Danyel, and Wigdor, Daniel (2010), 'Search on surfaces: Exploring the potential of interactive tabletops for collaborative search tasks', *Inf. Process. Manage.*, 46 (6), 703-17.

Morsella, Ezequiel and Krauss, Robert M. (2004), 'The Role of Gestures in Spatial Working Memory and Speech', *The American Journal of Psychology,* 117 (3), 411-24.

Moscovitch, M., Kapur, S., Kohler, S., and Houle, S (1995), 'Distinct neural correlates of visual long-term memory for spatial location and object identity: A position emission tomography study in humans', *Proc Natl Acad Sci U S A.,* 92 (9), 3721-25.

Müller-Prove, M. and Ludolph, Frank (2007), 'Dueling interaction models of personal-computing and web-computing', *Methodic and Didactic Challenges of the History of Informatics (MEDICHI 2007)* (Klagenfurt, Austria), 34-38.

Myers, B. A. (1989), 'Encapsulating interactive behaviors', *Proceedings of the SIGCHI conference on Human factors in computing systems: Wings for the mind* (ACM), 319-24.

Myers, Brad, Hudson, Scott E, and Pausch, Randy (2000), 'Past, present, and future of user interface software tools', *ACM Trans. Comput.-Hum. Interact.,* 7, 3-28.

Myers, Brad A (1998), 'A brief history of human-computer interaction technology', *interactions,* 5, 44-54.

Nacenta, Miguel A. (2008), 'Inventing the Future of Multi-Display Environments', *Beyond the Laboratory: Supporting Authentic Collaboration with Multiple Displays (a CSCW 2008 workshop)* (San Diego, CA, USA).

Nacenta, Miguel A., Gutwin, Carl, Aliakseyeu, Dzmitry, and Subramanian, Sriram (2009), 'There and Back Again: Cross-Display Object Movement in Multi-Display Environments', *Human–Computer Interaction,* 24 (1-2), 170-229.

Nielsen, Jakob (1993), *Usability Engineering (Interactive Technologies)* (Morgan Kaufmann).

Norman, D.A. (2002), *The Design of Everyday Things* (New York: Basic Books).

Novak, Joseph D. and Gowin, D. Bob (1984), *Learning How To Learn* (Cambridge, UK: Cambridge University Press).

Oleksik, Gerard, Milic-Frayling, Natasa, and Jones, Rachel (2012), 'Beyond data sharing: artifact ecology of a collaborative nanophotonics research centre', *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work* (Seattle, Washington, USA: ACM), 1165-74.

Oulasvirta, Antti (2008), 'When users "do" the Ubicomp', *interactions,* 15, 6.

Pawson, Richard (2002), 'Naked objects', *Software, IEEE,* 19, 81-83.

--- (2004), 'Naked objects', PhD Thesis (University of Dublin).

Pawson, Richard and Matthews, Robert (2001), 'Naked objects: a technique for designing more expressive systems', *ACM SIGPLAN Notices* (36: ACM), 61-67.

Pawson, Richard, Bravard, Jean-Louis, and Cameron, Lorette (1995), 'The Case for Expressive Systems', *MIT Sloan Management Review,* 36, 41-48.

Perlin, Ken and Fox, David (1993), 'Pad: an alternative approach to the computer interface', *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques* (New York, NY, USA: ACM Press), 57-64.

Perlin, Ken and Meyer, Jon (1999), 'Nested user interface components', *Proceedings of the 12th annual ACM symposium on User interface software and technology* (Asheville, North Carolina, United States: ACM), 11-18.

Pirolli, P. and Card, S. (2005), 'The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis', *Proceedings of International Conference on Intelligence Analysis*, 2--4.

Plaisant, Catherine (2004), 'The challenge of information visualization evaluation', *Proceedings of the working conference on Advanced visual interfaces* (Gallipoli, Italy: ACM), 109-16.

Plumlee, Matthew D and Ware, Colin (2006), 'Zooming versus multiple window interfaces', *ACM Transactions on Computer-Human Interaction,* 13, 179-209.

Rädle, Roman, Butscher, Simon, Huber, Stephan, and Reiterer, Harald (2012), 'Navigation Concepts for ZUIs Using Proxemic Interactions', *Proxemics '12 (a Workshop in conjunction with NordiCHI '12)* (Copenhagen: HCI Group, University of Konstanz).

Raskin, Jef (2000), *The humane interface : new directions for designing interactive systems* (Reading, Mass.: Addison-Wesley).

Ravasio, Pamela and Tscherter, Vincent (2007), 'User's Theories of the Desktop Metaphor or Why We Should Seek Metaphor-Free Interfaces', in Victor Kaptelinin and Mary Czerwinski (eds.), *Beyond the Desktop Metaphor : Desiging Integrated Digital Work Environments* (Cambridge, MA, USA: MIT Press), 265-94.

Reiterer, Harald (2011), 'Human-computer interaction group: University of Konstanz, Germany', *interactions,* 18 (6), 82-85.

Rekimoto, Jun and Saitoh, Masanori (1999), 'Augmented surfaces: a spatially continuous work space for hybrid computing environments', *Proceedings of the SIGCHI conference on Human Factors in Computing Systems* (Pittsburgh, Pennsylvania, United States: ACM), 378-85.

Rheingold, Howard (2000), *Tools for Thought: The History and Future of Mind-Expanding Technology* (MIT Press).

Roberts, Dave, Berry, Dick, Isensee, Scott, and Mullaly, John (1998), *Designing for the User with OVID: Bridging the Gap Between Software Engineering and User Interface Design* (Macmillan Technical Publishing).

Robertson, George, Czerwinski, Mary, Larson, Kevin, Robbins, Daniel C, Thiel, David, and Dantzich, Maarten Van (1998), 'Data Mountain : Using Spatial Memory for Document Management', *Proc. UIST '98* (San Francisco, California, United States: ACM), 153-62.

Robertson, George, Dantzich, Maarten van, Robbins, Daniel, Czerwinski, Mary, Hinckley, Ken, Risden, Kirsten, Thiel, David, and Gorokhovsky, Vadim (2000), 'The Task Gallery: a 3D window manager', *Proceedings of the SIGCHI conference on Human factors in computing systems* (The Hague, The Netherlands: ACM), 494-501.

Robinson, Mike (1993), 'Design for unanticipated use', *Proc. ECSCW '93* (New York, New York, USA: ACM Press), 187-202.

Robles, Erica, Nass, Clifford, and Kahn, Adam (2009), 'The Social Life of Information Displays: How Screens Shape Psychological Responses in Social Contexts', *Human–Computer Interaction,* 24 (1-2), 48-78.

Rogers, Yvonne and Lindley, Siân (2004), 'Collaborating around vertical and horizontal large interactive displays: which way is best?', *Interacting with Computers,* 16 (6), 1133-52.

Rogers, Yvonne, Lim, Youn-kyung, Hazlewood, William R., and Marshall, Paul (2009), 'Equal Opportunities: Do Shareable Interfaces Promote More Group Participation Than Single User Displays?', *Human–Computer Interaction,* 24 (1-2), 79-116.

Román, Manuel, Hess, Christopher, Cerqueira, Renato, Ranganathan, Anand, Campbell, Roy H., and Nahrstedt, Klara (2002), 'A Middleware Infrastructure for Active Spaces', *IEEE Pervasive Computing,* 1 (4), 74-83.

Runge, Oliver (2010), 'Design and Implementation of a Zoom and Facet-based Visualization', Bachelor's Thesis (University of Konstanz).

Ryall, Kathy, Forlines, Clifton, Shen, Chia, Morris, Meredith Ringel, and Everitt, Katherine (2006), 'Experiences with and Observations of Direct-Touch Tabletops', *Proceedings of the First IEEE International Workshop on Horizontal Interactive Human-Computer Systems* (IEEE Computer Society), 89-96.

Schmidt, Sebastian, Nacenta, Miguel A., Dachselt, Raimund, and Carpendale, Sheelagh (2010), 'A set of multi-touch graph interaction techniques', *ACM International Conference on Interactive Tabletops and Surfaces* (Saarbrücken, Germany: ACM), 113-16.

Schubert, Sören (2010), 'Auswirkung verschiedener Eingabegeräte auf das Erlernen räumlich-visueller Merkmale einer virtuellen Informationslandschaft', Master's Thesis (University of Konstanz).

Schweizer, Jonas (2009), 'Gestaltung und Implementierung eines Multi-Fokus und Multi-Display Management-Systems', Bachelor's Thesis (University of Konstanz).

Scott, Stacey (2005), 'Territoriality in Collaborative Tabletop Workspaces', PhD Thesis (University of Calgary).

Scott, Stacey D., Carpendale, Sheelagh, and Inkpen, Kori M. (2004), 'Territoriality in collaborative tabletop workspaces', *Proceedings of the 2004 ACM conference on Computer supported cooperative work* (Chicago, Illinois, USA: ACM), 294-303.

Seifried, Thomas, Jetter, Hans-Christian, Haller, Michael, and Reiterer, Harald (2011), 'Lessons Learned from the Design and Implementation of Distributed Post-WIMP User Interfaces', in José A. Gallud, Ricardo Tesoriero, and Victor M. R. Penichet (eds.), *Distributed User Interfaces: Designing Interfaces for the Distributed Ecosystem* (Human-Computer Interaction Series: Springer London), 95-102.

Seifried, Thomas, Rendl, Christian, Haller, Michael, and Scott, Stacey (2012), 'Regional undo/redo techniques for large interactive surfaces', *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems* (Austin, Texas, USA: ACM), 2855-64.

Shaer, Orit and Jacob, Robert J K (2009), 'A specification paradigm for the design and implementation of tangible user interfaces', *ACM Transactions on Computer-Human Interaction,* 16, 1-39.

Shen, Chia, Vernier, Frédéric D., Forlines, Clifton, and Ringel, Meredith (2004), 'DiamondSpin: an extensible toolkit for around-the-table interaction', *Proceedings of the SIGCHI conference on Human factors in computing systems* (Vienna, Austria: ACM), 167-74.

Shen, Chia, Lesh, Neal B., Vernier, Frederic, Forlines, Clifton, and Frost, Jeana (2002), 'Building and sharing digital group histories', *Proceedings of the 2002 ACM on Computer supported cooperative work video program* (New Orleans, Louisiana: ACM), 3-3.

Shih, C.C., Crone, M., Fox, Armando, and Winograd, T. (2004), 'Teamspace: A Simple, Low-Cost and Self-Sufficient Workspace for Small-Group Collaborative Computing (Interactive Poster)', *CSCW 2004* (Chicago, IL, USA: ACM Press).

Shneiderman, Ben (1982), 'The future of interactive systems and the emergence of direct manipulation', *Behaviour & Information Technology*, 1, 237-56.

--- (1983), 'Direct manipulation: a step beyond programming languages', *IEEE Computer*, 57-69.

--- (1996), 'The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations', *Proceedings of the 1996 IEEE Symposium on Visual Languages* (IEEE Computer Society), 336.

--- (1997), *Designing the User Interface: Strategies for Effective Human-Computer Interaction* (Addison-Wesley Longman Publishing Co., Inc.) 639.

--- (2002), *Leonardo's Laptop: Human Needs and the New Computing Technologies* (MIT Press) 256.

Shupp, Lauren, Andrews, Christopher, Dickey-Kurdziolek, Margaret, Yost, Beth, and North, Chris (2009), 'Shaping the Display of the Future: The Effects of Display Size and Curvature on User Performance and Insights', *Human–Computer Interaction*, 24 (1-2), 230-72.

Smith, David A., Raab, Andreas, Reed, David P., and Kay, Alan (2004), 'Croquet: A Menagerie of New User Interfaces', *Proceedings of the Second International Conference on Creating, Connecting and Collaborating through Computing* (IEEE Computer Society), 4-11.

Søgaard, Mads (2006), 'Object orientation redefined: from abstract to direct objects and toward a more understandable approach to understanding', *Interaction-Design.org* <http://www.interaction-design.org/mads/articles/object_orientation_redefined.html>, accessed Aug 23, 2011.

Streitz, Norbert, Tandler, Peter, Müller-Tomfelde, Christian, and Konomi, Shinichi (2001), 'Roomware: Towards the Next Generation of Human-Computer Interaction based on an Integrated Design of Real and Virtual Worlds', in John M Carroll (ed.), *Human-Computer Interaction in the New Millenium* (Addison Wesley), 551-76.

Streitz, Norbert, Geißler, Jörg, Holmer, Torsten, Konomi, Shin'ichi, Müller-Tomfelde, Christian, Reischl, Wolfgang, Rexroth, Petra, Seitz, Peter, and Steinmetz, Ralf (1999), 'i-LAND', *Proc. CHI '99* (New York, New York, USA: ACM Press), 120-27.

Suchman, Lucy A (1987), *Plans and situated actions: the problem of human-machine communication* (New York, NY, USA: Cambridge University Press).

Tan, Desney S., Pausch, Randy, Stefanucci, Jeanine K., and Proffitt, Dennis R. (2002), 'Kinesthetic cues aid spatial memory', *CHI '02 extended abstracts on Human factors in computing systems* (Minneapolis, Minnesota, USA: ACM), 806-07.

Tandler, Peter, Prante, Thorsten, Müller-Tomfelde, Christian, Streitz, Norbert, and Steinmetz, Ralf (2001), 'Connectables: dynamic coupling of displays for the flexible creation of shared workspaces', *Proceedings of the 14th annual ACM symposium on User interface software and technology* (Orlando, Florida: ACM), 11-20.

Tang, Anthony and Fels, Sidney (2008), 'Four lessons from traditional MDEs', in Jacob T. Biehl, Gene Golovchinsky, and Kent Lyons (eds.), *ACM CSCW 2008 Workshop on Beyond the Laboratory: Supporting Authentic Collaboration with Multiple Displays*.

Tang, Anthony, Tory, Melanie, Po, Barry, Neumann, Petra, and Carpendale, Sheelagh (2006), 'Collaborative coupling over tabletop displays', *Proceedings of the SIGCHI conference on Human Factors in computing systems* (Montreal, Quebec, Canada: ACM), 1181-90.

Taylor, Stuart, Izadi, Shahram, Kirk, David, Harper, Richard, and Garcia-Mendoza, Armando (2009), 'Turning the tables: an interactive surface for vjing', *Proceedings of the 27th international conference on Human factors in computing systems* (Boston, MA, USA: ACM), 1251-54.

Terrenghi, Lucia, Quigley, Aaron, and Dix, Alan (2009), 'A taxonomy for and analysis of multi-person-display ecosystems', *Personal Ubiquitous Comput.,* 13 (8), 583-98.

Tesler, Larry (1983), 'Object-oriented user interfaces and object-oriented languages (Keynote Address)', *Proceedings of the 1983 ACM SIGSMALL symposium on Personal and small computers* (San Diego, California, United States: ACM), 3-5.

Thomas, James J. and Cook, Kristin A. (2005), *Illuminating the path : the research and development agenda for visual analytics* (1st edn.: National Visualization and Analytics Ctr).

Tibbetts, John (1991), 'Object orientation and transaction processing: where do they meet?', *SIGPLAN OOPS Mess.,* 3 (4), 3-15.

Ullmer, Brygg and Ishii, Hiroshi (1997), 'The metaDESK: models and prototypes for tangible user interfaces', *Proceedings of the 10th annual ACM symposium on User interface software and technology* (Banff, Alberta, Canada: ACM), 223-32.

Ullmer, Brygg, Ishii, Hiroshi, and Jacob, Robert J. K. (2003), 'Tangible Query Interfaces: Physically Constrained Tokens for Manipulating Database Queries', *Proceedings of INTERACT '03*, 279-86.

van Asselen, Marieke (2005), 'The neurocognitive basis of spatial memory', PhD Thesis (Helmholtz Institute, Utrecht University).

van Dam, Andries (1997), 'Post-WIMP user interfaces', *Communications of the ACM,* 40, 63-67.

Van Wijk, Jarke J. and Nuij, Wim A. A. (2003), 'Smooth and efficient zooming and panning', *Proceedings of the Ninth annual IEEE conference on Information visualization* (Seattle, Washington: IEEE Computer Society), 15-22.

Voida, Stephen, Mynatt, Elizabeth D., and Edwards, W. Keith (2008), 'Re-framing the desktop interface around the activities of knowledge work', *Proceedings of the 21st annual ACM symposium on User interface software and technology* (Monterey, CA, USA: ACM), 211-20.

Vyas, Dhaval, Veer, Gerrit, Heylen, Dirk, and Nijholt, Anton (2009), 'Space as a Resource in Creative Design Practices', *Proceedings of the 12th IFIP TC 13 International Conference on Human-Computer Interaction: Part II* (Uppsala, Sweden: Springer-Verlag), 169-72.

Wagner, Kirsten (2000), 'Informations- und Wissensorganisation anhand räumlicher Ordnungsmodelle. Das Spatial Data-Management System der Architecture Machine Group als Fallbeispiel.', *Wolkenkuckucksheim. Internationale Zeitschrift für Theorie und Wissenschaft der Architektur*.

Ware, Colin (2004), *Information visualization: perception for design* (San Francisco, CA, USA: Morgan Kaufmann Publishers Inc).

Weiser, M (1991), 'The computer for the 21st century', *Scientific American,* 3, 94-104.

Wesp, Richard, Hesse, Jennifer, Keutmann, Donna, and Wheaton, Karen (2001), 'Gestures Maintain Spatial Imagery', *The American Journal of Psychology,* 114 (4), 591-600.

Wexler, M., Kosslyn, S., and Berthoz, A. (1998), 'Motor processes in mental rotation', *Cognition,* 68, 77-94.

Wigdor, Daniel and Wixon, Dennis (2011), *Brave NUI World : designing natural user interfaces for touch and gesture* (Morgan Kaufmann) 264.

Wigdor, Daniel, Shen, Chia, Forlines, Clifton, and Balakrishnan, Ravin (2006), 'Table-Centric Interactive Spaces for Real-Time Collaboration', *World*, 103-07.

Wobbrock, Jacob O., Morris, Meredith Ringel, and Wilson, Andrew D. (2009), 'User-defined gestures for surface computing', *Proceedings of the 27th international conference on Human factors in computing systems* (Boston, MA, USA: ACM), 1083-92.

Yee, Ka-Ping (2003), 'Peephole displays: pen interaction on spatially aware handheld computers', *Proceedings of the SIGCHI conference on Human factors in computing systems* (Ft. Lauderdale, Florida, USA: ACM), 1-8.

Yee, Ka-Ping, Swearingen, Kirsten, Li, Kevin, and Hearst, Marti (2003), 'Faceted metadata for image search and browsing', *Proceedings of the SIGCHI conference on Human factors in computing systems* (Ft. Lauderdale, Florida, USA: ACM), 401-08.

Young, Degi and Shneiderman, Ben (1993), 'A graphical filter/flow representation of Boolean queries: a prototype implementation and evaluation', *J. Am. Soc. Inf. Sci.*, 44 (6), 327-39.

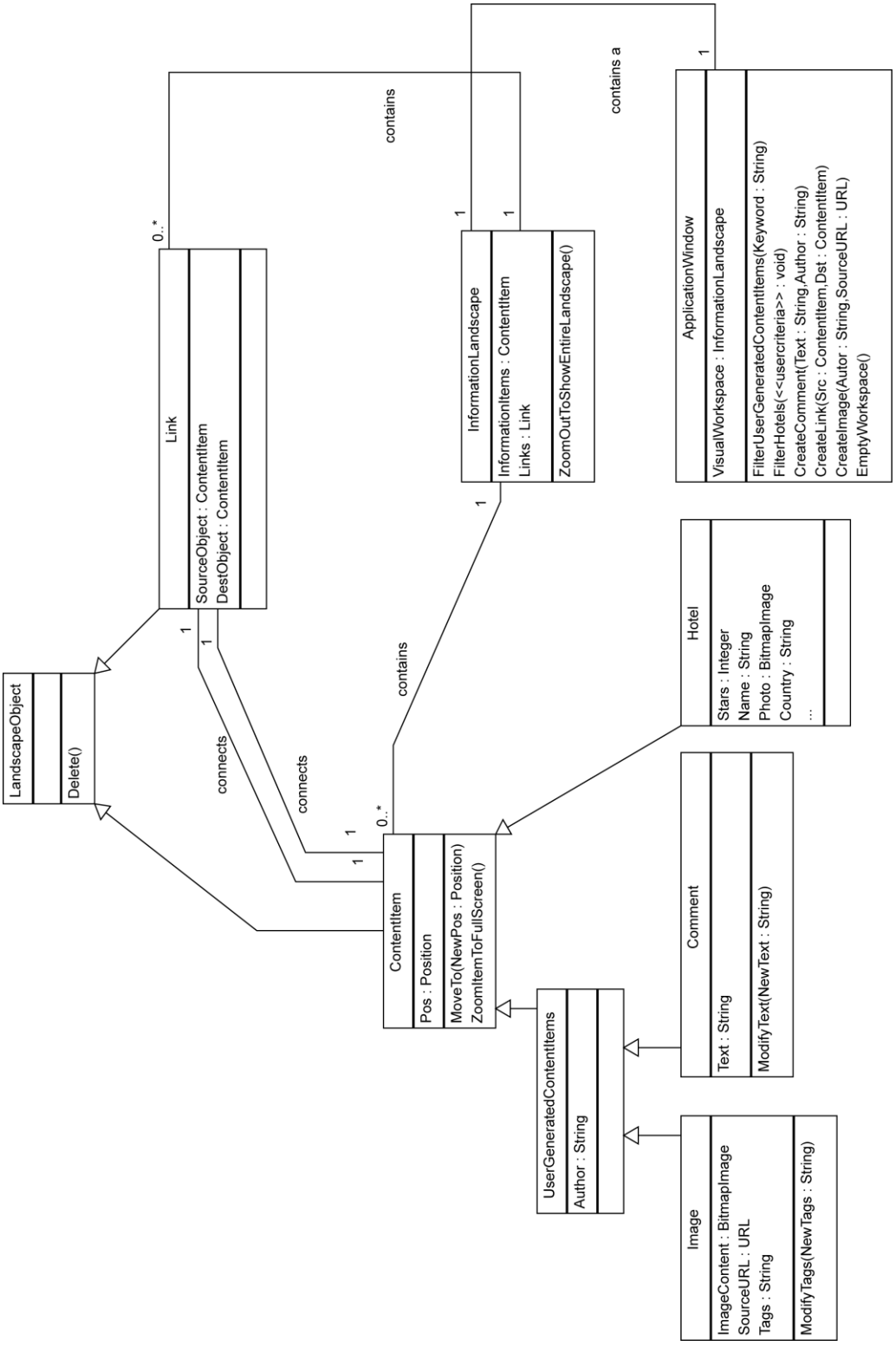
Zöllner, Michael (2009), 'Ein persistentes objektorientiertes Datenmodell für das Personal Information Management in ZOIL', Bachelor's Thesis (University of Konstanz).

--- (2012), 'A state machine framework for Post-WIMP interaction design', Master's Thesis (University of Konstanz).

Zöllner, Michael, Jetter, Hans-Christian, and Reiterer, Harald (2011), 'ZOIL: A Design Paradigm and Software Framework for Post-WIMP Distributed User Interfaces', in José A. Gallud, Ricardo Tesoriero, and Victor M. R. Penichet (eds.), *Distributed User Interfaces: Designing Interfaces for the Distributed Ecosystem* (Human-Computer Interaction Series: Springer London), 87-94.

9 Appendix

9.1 Appendix for Section 3.2



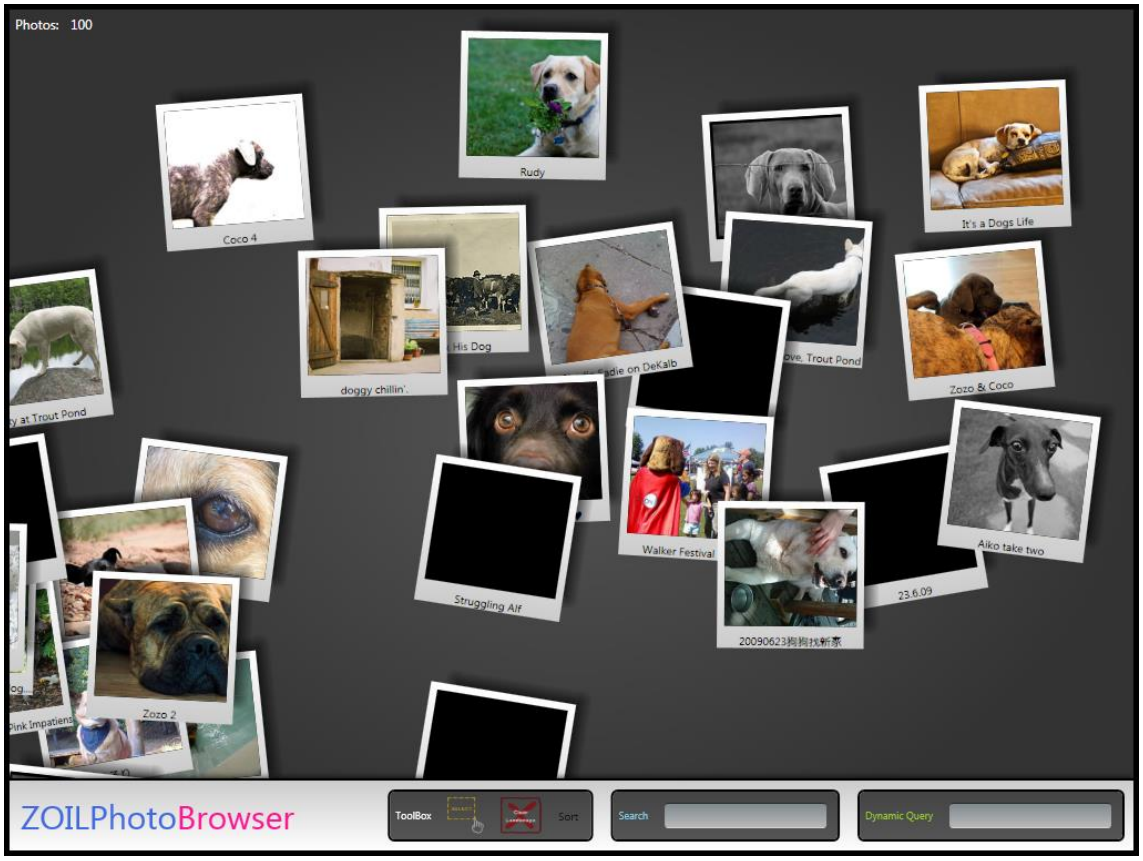
Solution model for OOUI case study from section 3.2.

9.2 Appendix for Section 3.4

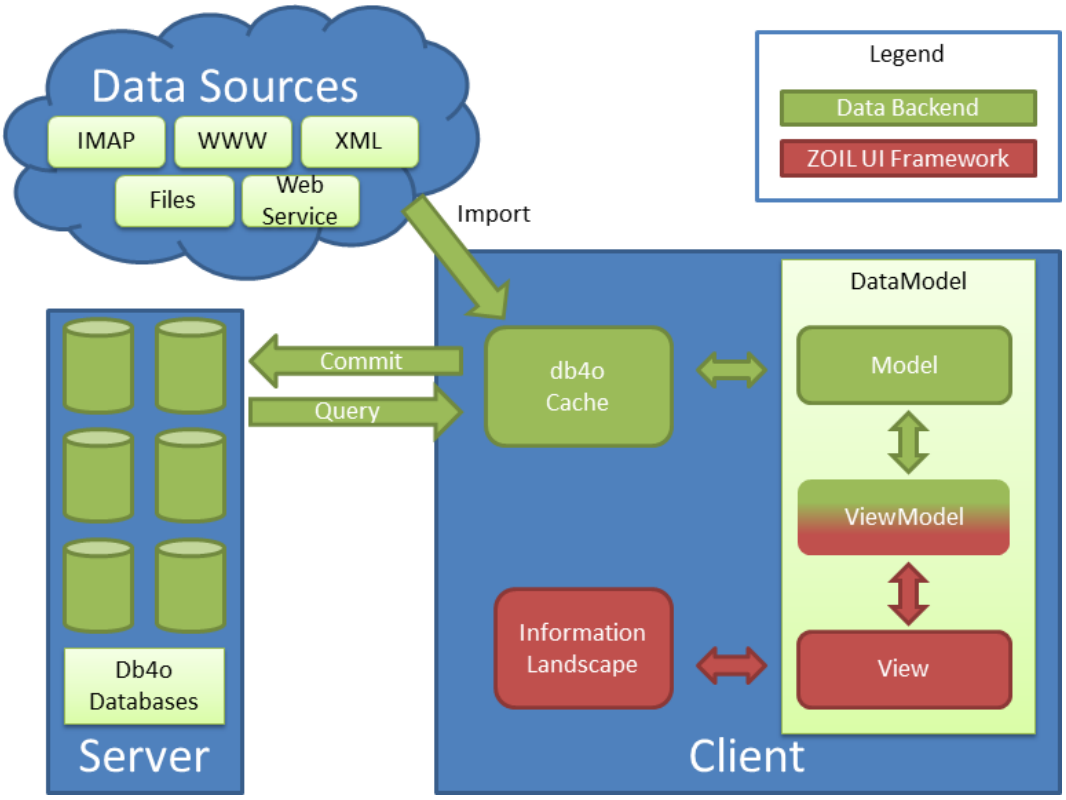
The most recent public version of the ZOIL software framework can be downloaded as an open source project under the BSD license from *http://zoil.codeplex.com*.

To get started with the framework, I recommend watching the video recordings of our lectures and tutorials on how to use the ZOIL software framework in own projects (German language, length 4 hours): *http://streaming.uni-konstanz.de/talks-events/zoil-workshop/*.

A good starting point for learning ZOIL is the ZOIL PhotoBrowser sample from the folder *Samples/PhotoBrowser/Sample.PhotoBrowser* of the ZOIL framework distribution. It demonstrates how to create an information landscape and to share it via the ZOIL server, how to populate the landscape with images from the Flickr web service, and how to define different size-dependent representations of Flickr images for semantic zooming. Further contained functions are filtering images by name, selections by drawing shapes and bounding boxes around images, and displaying simple visualizations.



Screenshot of the ZOIL PhotoBrowser Sample.



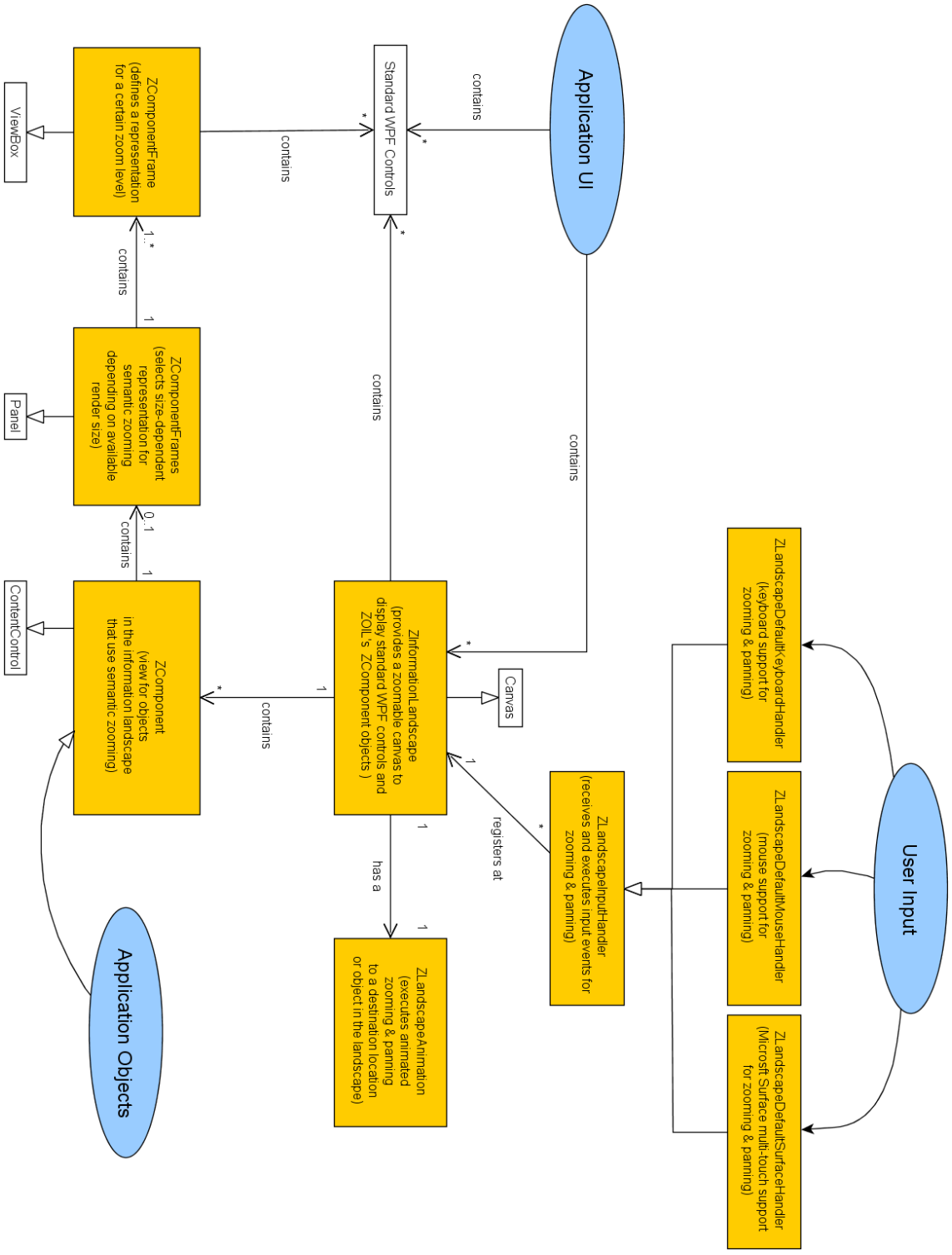
Interplay between the different components of the ZOIL Software Framework.

The ZOIL PhotoBrowser Sample is an example for using the ZOIL framework's architecture from the figure above. The sample uses the client-side data backend components ("Model", "db4o Cache") to connect to the ZOIL server that contains one or several ZOIL databases. Each database on the server stores a binary representation of the data model of a single information landscape.

The ZOIL client uses db40 queries and commits ("Query", "Commit") to continuously synchronize the data model of the information landscape on the client-side ("Model") with that stored on the server. This happens automatically and transparently to the programmer using the *Transparent Persistence* pattern of db4o. Thereby the "db4o Cache" component reduces the amount of necessary commits and queries on the database by collecting smaller changes to execute them in bulks and locally caching the results of executed queries. This can reduce network and database load when the same query is repeated many times.

The *Model-View-ViewModel* pattern in the ZOIL framework continuously visualizes and updates the content of the information landscape in the client's UI. Using two-way data binding, the changes to the data model are automatically detected and become instantly visible on the UI and user manipulations of objects on the UI are instantly applied to the local data model and transparently persisted on the server.

In the case of the ZOIL PhotoBrowser, the Flickr web service serves as a data source. After sending keyword queries to the Flickr web service, the data source asynchronously populates the data model with the resulting images which then appear on the UI.



Architecture of the presentation and interaction components of the ZOIL framework.

Above figure gives an overview of the most essential classes and their relations from the ZOIL software framework that are used when implementing the presentation layer of an application like the PhotoBrowser. In the following, I briefly describe their different roles:

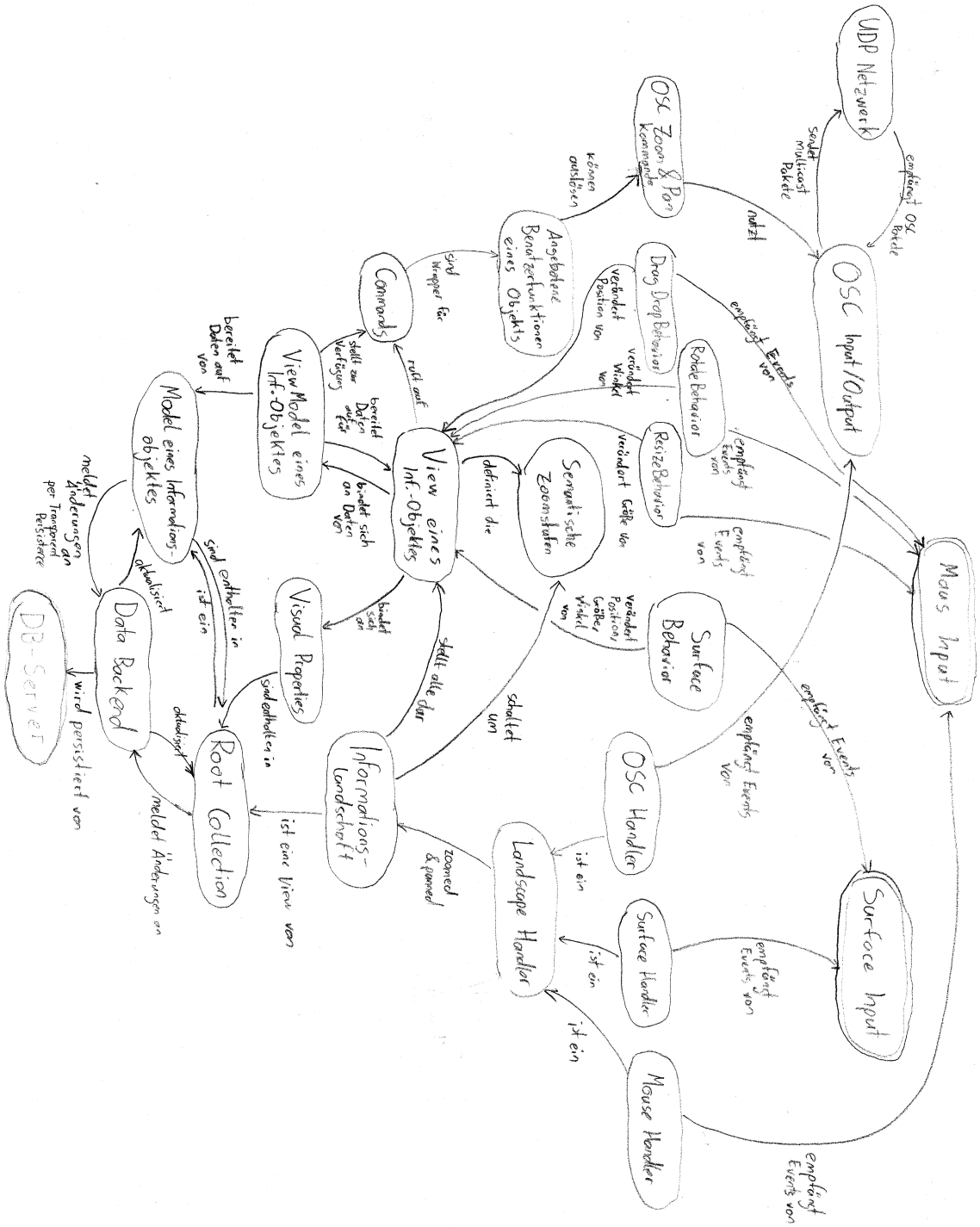
A ZOIL application's user interface can be composed of arbitrary WPF controls, e.g., the "ZOILPhotoBrowser" label and the global controls and text entry fields shown at the bottom of the screenshot of the PhotoBrowser sample. To host ZOIL's information landscape, an application uses one or many instances of a *ZInformationLandscape* object. This object is derived from WPF's *Canvas* object and hosts a viewport that renders a region of a zoomable information landscape. In the screenshot of the PhotoBrowser, the entire top section of the screen is covered by this *ZInformationLandscape* that enables navigating and manipulating application's domain objects (in this case images from Flickr that are shown as Polaroid pictures).

Users can zoom and pan in the information landscape with different input devices. The public version of the ZOIL framework currently contains keyboard, mouse, or multi-touch support (*ZLandscapeDefaultKeyboardHandler*, *ZLandscapeDefaultMouseHandler*, *ZLandscapeDefaultSurfaceHandler*). Own input device handlers can be easily implemented by inheriting from the *ZLandscapeInputHandler* base class, for example, to support Windows 7 multi-touch events or the Nintendo Wiimote controller. After registering a *ZLandscapeInputHandler* object at the *ZInformationLandscape* object, the landscape can be zoomed and panned with the input device. To achieve this, the handler objects internally access the *ZLandscapeAnimation* object of the *ZInformationLandscape* to access functions for jumping or smoothly moving to objects or arbitrary regions and scales in the landscape.

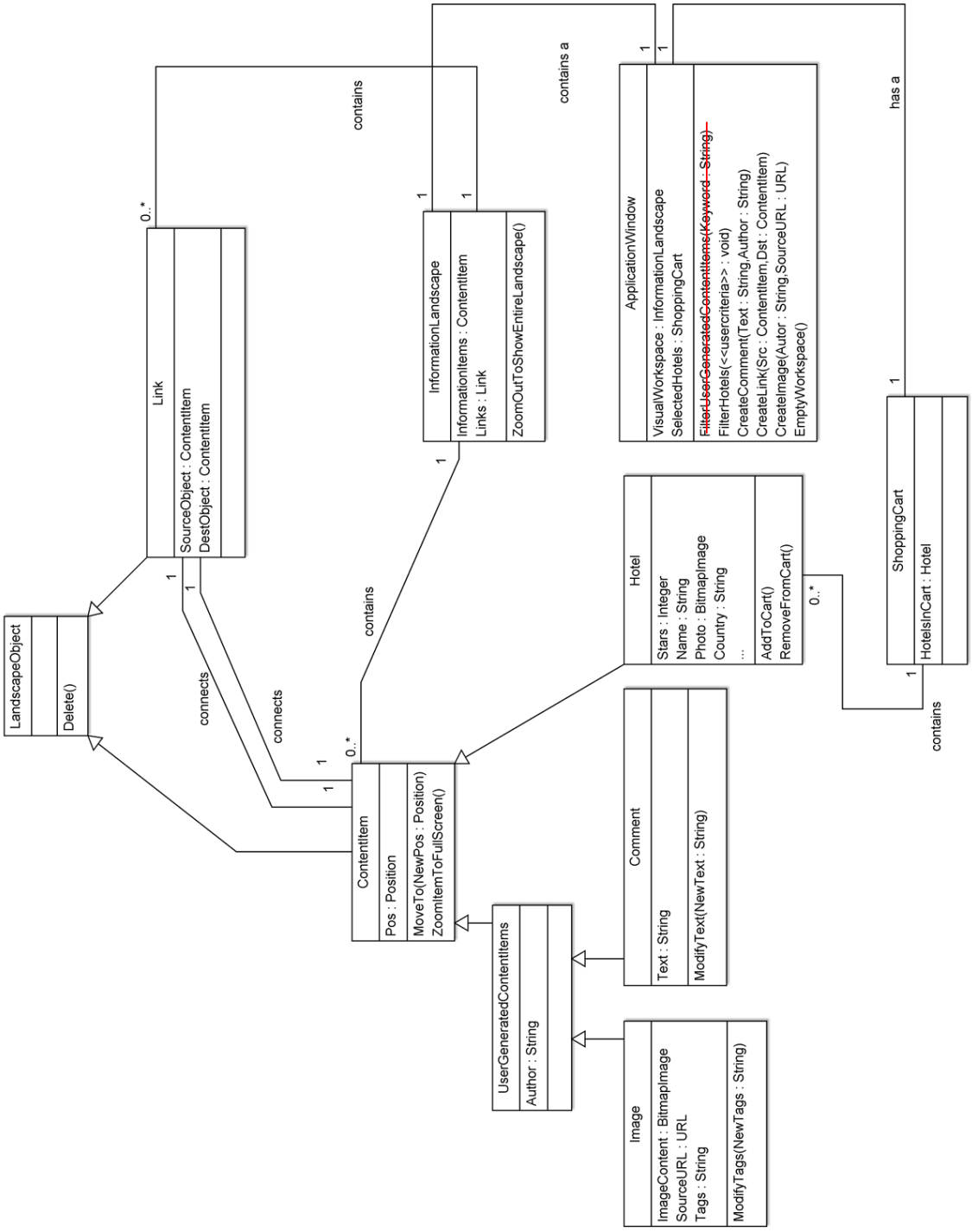
The key feature of ZOIL's presentation layer is its support for semantic zooming that is based on the *ZComponent* object. Typically, the *ZComponent* object is used to visually instantiate a domain object (e.g., an image, a note, a hotel) in the information landscape. To implement new objects for semantic zooming in the information landscape, the application creates a new class for a domain object (e.g., *FlickrImage*) that inherits from *ZComponent*.

Each *ZComponent* object contains a *ZComponentFrames* object that contains different size-dependent views, i.e., the different *ZComponentFrame* objects, and shows them according to the currently available render size for the *ZComponent*. The actual content and the controls for each *ZComponentFrame* are typically defined using XAML (see section 4.3.5).

9.3 Appendix for Section 3.5



Concept map of the ZOIL Framework created by the creators of the framework during the API usability evaluation study.



Specification of the Hotel-Browser UI from the 1st API usability evaluation study.