BACHELORARBEIT

A Design Space for User Interface Design in Collaborative Systems

Zur Erlangung des akademischen Grades
Bachelor of Science
(BSc.)

ausgeführt am
Institut für Rechnergestützte Automation
Forschungsgruppe Industrial Software

der Technischen Universität Wien

unter der Anleitung von
Univ.-Prof. Dipl.-Ing. Dr. Thomas Grechenig
und
Dipl.-Ing. Johannes Harms

durch: Christoph Derndorfer
Rauscherstrasse 8 / 34, 1200 Wien

Wien, April 2013

Kurzfassung

IKT-basierte Kollaborationssysteme wurden, und werden weiterhin, in vielen gesellschaftlichen Bereichen immer wichtiger. Allerdings bleibt das Design ihrer Benutzeroberflächen, aufgrund der vielen Möglichkeiten und wachsenden Anforderungen an Kollaboration über verschiedene Nutzungskontexte und Geräte hinweg, weiterhin herausfordernd. Um Entscheidungen in der Analyse von bestehenden und dem Design von zukünftigen Kollaborationssystemen zu unterstützen, wurde ein Design Space von Optionen für Benutzeroberflächen entwickelt. Der Design Space basiert auf einer systematischen Analyse von aktuellen Kollaborationssystemen und deren Designoptionen. Dazu wurde eine anfängliche Auswahl von Designoptionen aus bestehender Literatur iterativ mit neuen Optionen, die durch einen Review von fünf Kollaborationssystemen identifiziert wurden, erweitert. Im Gegensatz zu anderen Design Spaces oder Klassifikationen konzentriert sich dieser Design Space auf Optionen für Benutzeroberflächen, wurde basierend auf Reviews von öffentlich verfügbaren Systemen erstellt und berücksichtigt den mobilen Einsatz von Kollaborationssystemen. Der Design Space wird besonders in den frühen Phasen des Software-Design-Prozess als besonders nützlich erachtet.

Abstract

ICT based collaboration systems have, and continue to, become increasingly important in many sectors of society. However the design of their user interfaces remains challenging due to the many possible options and the increasing demands of collaborating across different usage contexts and devices. To support decisions in the analysis of current and design of future collaborative systems a design space of user interface design options was developed. It is based on a systematic analysis of current collaborative products and their design options. An initial set of design options from existing literature was iteratively extended with additional options that were identified through a review of five collaboration systems. Contrary to other design spaces and classifications this design space puts a narrower focus on user interface design options, reviewed publicly available systems, and includes considerations relevant in the mobile use of collaboration software. In terms of its application this design space is believed to be particularly relevant during the early stages of the software design process.

Schlüsselwörter

Kollaboration, Design Space, User Interface Design, Groupware, CSCW.

Keywords

collaboration, design space, user interface design, groupware, CSCW.

Table of Contents

1 Introduction

Collaboration with the support of ICT based systems has become increasingly popular in the last few years, particularly when it comes to the collaborative production of texts and other text based artifacts. Today these collaborative workflows can be found in news publications, hospitals, software companies, government organizations, and many other sectors of society.

This collaboration is enabled by a broad variety of ICT technologies, platforms, and products. These range from traditional approaches such as chats and e-mails to newer solutions such as Google Docs, wikis, and Etherpad. At the same time there is a broad variety of domain-specific applications and solutions, for example version control systems which are widely used in software development.

Within this context, this thesis focuses on user interface design. Its purpose is to provide a design space for the analysis of current and design of future collaborative systems. The method is based on a review of existing products. In contrast to other works, options for user interface design rather than the products themselves are classified. Due to its purpose and structure, the classification can also be interpreted as 'design space' [MacLean u. a., 1996]. In this spatial metaphor, the classification's categories are understood as dimensions spanning up an imaginary space of design options.

1.1 Related work

Classifications and taxonomies have been employed to organize and analyze collaborative products and services for more than thirty years. A comprehensive overview of such efforts is provided in [Penichet u. a., 2007]. As the paper puts it: "Taxonomies provide a way to classify different groupware tools." Essentially these classifications and taxonomies focused on what kind of collaboration a system supported, often built on the time and space matrix introduced in [Johansen, 1988]. Additionally some researchers emphasized the notion of different processes taking place in collaboration (e.g., communication and coordination). The new classification proposed in [Penichet u. a., 2007] went a step further by essentially combining both of these approaches. This thesis builds on that prior work by using Johansen's time and space approach to describe the usage context of the collaboration products which are analyzed. The goal for the resulting classification is to help organize options for user interface design rather than organize the products themselves.

Another related paper which influenced the classification approach in this thesis is [Rama und Bishop, 2006]. Among other things the paper includes a set of criteria which was selected to enable a comparison between existing collaboration systems and the system developed by the researchers. The focus was therefore again set on differentiating collaboration systems. However its two-level approach led to the organization of the user interface design options into different classification categories used in this thesis.

A recent example which indicated that classification can also be applied beyond the comparison of different types of collaboration systems was encountered in [Hincapié-Ramos u. a., 2011]. In it a design space was developed to analyse so-called availability-sharing systems. Beyond the mere analyses of existing systems the design space also aimed to provide a foundation "to identify previously unexplored points in the design space, for designing new systems". Subsequently that approach was used to design a system with a unique combination of characteristics. This narrow focus, a design space which can be used in a retrospective way to analyze current availability-sharing systems and a forward-thinking fashion to inform the design of new systems, ultimately inspired the approach used in this thesis. However a significant difference is that this thesis proposes a design space which can be used for any type of collaboration system.

1.2 Structure of this thesis

The next chapter 2 contains a brief overview of the development of collaboration systems in the past 40 to 50 years. It is followed by chapter 3 which explains the methods used. Chapter 4 focuses on the design space developed in this thesis. Chapter 5 contains the reviews of various products and services. This is followed by an overview of the main findings and associated explanations in chapter 6. The conclusions are presented in chapter 7.

2 History of Text-Based Collaboration Systems

This chapter briefly outlines the development of text-based collaborative systems from its first steps to modern systems commonly used today. It also highlights notable efforts in related research fields. By describing past developments this chapter hopes to inspire readers to re-examine prior research, particularly with regard to its early focus on the contexts of collaboration. That research provides a wealth of knowledge useful in the analysis and design of collaborative systems today. As such it is a good complement of the design space presented in this thesis.

Early days: The first text-based collaboration on computers started to appear in the 1960s. It was initially based around systems which supported multiple users logging into a central terminal and simply leaving text messages for each other in text files.

One of the first steps to faciliate this communication process was the 'mail' command implemented at MIT's Compatible Time-Sharing System (CTSS) in late 1964 or early 1965. The command was described as "a facility that would allow any CTSS user to send a message to any other"[1]. It is important to point out that initially this communication tool and other similar ones were limited to being able to exchange messages between users who were using or at least connected to the same physical computer.

Then in 1968 Douglas Engelbart gave a conference presentation called "A Research Center for Augmenting Human Intellect"which today is most widely known for its introduction of the mouse. Colloquially it is often also referred to as "The Mother of All Demos". Aside of introducing the mouse and other technologies which are now widely used Engelbart also presented what he called "joint-file usageänd "two-person collaboration". The system he demonstrated and described[2] contains many of the elements which are found in text-based collaboration systems today and will be explored in more detail in chapter 4.

It is interesting to note that both electronic mail and document-based collaboration systems were initiated at universities and research facilities. On the one hand this is a result of these institutions being the first organizations to actually have access to computers and networks. On the other hand it also indicates that there was a strong interest in and demand for text-based collaboration systems as they facilitated the work done in these academic settings.

Emergence of Groupware and CSCW: Approximately 10 years after Engelbart's presentation the topic of text-based collaboration systems started to receive more attention from researchers. One important date for these research efforts was October 4, 1978 when the expression "groupware"was first used by Peter and Trudy Johnson-Lenz. In a 1981 paper they subsequently defined it as "intentional group processes plus software to support them"[Johnson-Lenz und Johnson-Lenz, 1998].

[1] Van Vleck, Tom: The History of Electronic Mail. http://www.multicians.org/thvv/mail-history.html, 01.02.2001 (Last access: 30.09.2011)

[2] A video recording of the complete lecture as well as a version edited into 35 segments can be found at `http://sloan.stanford.edu/MouseSite/1968Demo.html`. Clips 23 and 25 contain a short demonstration and explanation of the collaboration mode.

Then in 1984 the term "Computer-Supported Cooperative Work" (CSCW) was born at a workshop organized by Irene Greif and Paul Cashman at MIT. One widely-cited paper describes the participants of that workshop as "people from various disciplines who shared an interest in how people work, with an eye to understanding how technology could support them". They coined the term CSCW to describe this new area of research[Grudin, 1994].

In 1986 that first workshop was followed by the significantly larger CSCW'86 conference and the subsequent 1988 publication of "Computer-supported cooperative work: a book of readings". The 793-page book edited by Greif represented an extensive collection of related research efforts. In it she also defined CSCW as "an identifiable research field focused on the role of the computer in group work" [Greif, 1988]. It is noteworthy that three out of the nine papers in part one of the book, called "Visions and First Steps Towards CSCW", were contributed by Douglas Engelbart. This demonstrates the strong influence he and his group's work and in particular his 1968 presentation had had on the field.

Overall this series of early events turned out to be very influential for collaboration systems for a long time. The distinction between groupware and CSCW as already indicated in the first definitions of these terms largely remains in effect today. Groupware still describes technical solutions in the form of software - and increasingly hardware - to enable and facilitate collaboration between people. CSCW is more broadly understood as the interdisciplinary research field that looks both at social and technological factors. Its goal is a better understanding of how people collaborate by for example looking at their usage context and figuring out how groupware fits into the picture.

One other key point to consider is that groupware and CSCW were - and largely still are - very much focused on work and work environments. Already during the opening remarks of the aforementioned CSCW'86 conference four different types of work were identified as being included in CSCW: authoring, research, design, and office work.[3].

As previously mentioned this focus can be attributed to the fact that even though personal computers had become more widespread throughout the 1980s the overwhelming majority of them were not connected to any networks. The Internet that existed at the time was only available to the military, academic and research institutions, and large corporations. The World Wide Web as we know it today had not even been invented yet. As such it was only natural for researchers and developers in academia and industry to focus on the types of work-related collaboration commonly found in their own organizations.

Towards broader adoption: The approximately 15 years between the late 1980s and the early 2000s can be characterized as text-based collaboration systems and similar groupware systems slowly becoming more widely available. Also its user base became more diverse via access by the general

[3]Nielsen, Jakob: CSCW'86 Trip Report. `http://www.useit.com/papers/tripreports/cscw86.html`. (Last access: 02.10.2011)

public which was not affiliated with the aforementioned organizations these systems were initially limited to.

Broader adoption and also increased diversity with regard to the development of different solutions then accelerated in tandem with the rapidly increasing number of Internet users in the first half of the 2000s. It was during this time frame that many text-based collaboration products and online services which are still widely used today started to be developed. Some of the most notable examples are MediaWiki, the software which powers Wikipedia, Writerly which later became the basis for Google Docs when Google acquired its original developers, and SubEthaEdit, a popular collaborative text editor for Mac OS X.

Current and future developments in CSCW: In the past few years one of the most notable developments within the CSCW field have been discussions about just how work-focused it should be. A number of papers have been published on this topic with one of the most regularly cited ones being a 2005 study by Crabtree et al. called "Moving with the Times: IT Research and the Boundaries of CSCW". In it the authors essentially argue that the research field's "horizon should be broadened to take in new developments in computing" [Crabtree u. a., 2005].

This significant step towards a broader horizon was also exemplified on the Web site of the CSCW 2010 conference: "Although work is an important area of focus for the conference, technology is increasingly supporting a wide range of recreational and social activities, and we embrace the growth of the discipline in the areas of Collaboration, Sociality, Computation, and the Web."[4]

It is interesting to note that in two ways the Web site for the recent 2012 edition of the CSCW conference went even further: "Although work is an area of focus, CSCW embraces research and technologies supporting a wide range of recreational and social activities using a diverse range of devices."[5] First of all work is no longer designated an *important* area of focus and secondly the explicit mention of *a diverse range of devices* clearly demonstrates that collaboration has moved beyond the confines of the traditional computer.

One recent example which illustrates the increasing importance of such these new devices is the Google Docs Android app. Google Docs itself is one of the most popular online office suites which was originally designed to be used with Web browsers on computers or laptops. With its Android app the service is also optimized for use with smartphones, and tablets[6].

So although it is essentially the same online service, access via the Web browser compared to the Android app differ significantly both in terms of a groupware and CSCW perspective. On the one hand the constraints of mobile devices, especially with regard to screen real estate and text input

[4]Luther, Kurt: CSCW 2010: The 2010 ACM Conference on Computer Supported Cooperative Work. `http://www.cscw2010.org/` (Last access: 1.10.2011)

[5]CSCW 2012: The 2012 ACM Conference on Computer Supported Cooperative Work. `http://www.cscw2012.org/` (Last access: 1.10.2011)

[6]Loxton, David: An enhanced Google Docs experience on mobile tablets. `http://googledocs.blogspot.com/2011/10/enhanced-google-docs-experience-on.html`, 05.10.2011 (Last access: 9.10.2011)

mechanisms, have to be considered. On the other hand the usage context of accessing the service on a mobile phone or tablet is likely also quite different from it being used on a computer or laptop.

Looking at these developments it is safe to assume that text-based collaboration systems and the broader fields of groupware and CSCW will continue to change and evolve over the coming years. This also provides ample opportunities for work related to the design, implementation, or research of such systems.

One particularly interesting challenge will be how to create systems which are accessible and usable in a variety of different usage contexts and on different devices. By providing an overview of the current challenges and solutions in text-based collaboration systems this thesis hopes to be able to contribute to these efforts.

3 Methods Used

The main research methods used to identify the design options for collaborative systems presented in this thesis are:

- Literature research
- Analyses and reviews of existing text-based collaboration products and services
- The design of a classification scheme

It is also important to point out that the order of the chapters in this thesis does not strictly follow the sequence of how the actual work was conducted. Rather the order was chosen to make the thesis easily accessible for readers with and without prior knowledge of the subject matter.

Initial analysis: The initial analysis of the existing products and services was intended to encompass a broad selection of different applications designed for a variety of usage contexts. It was also considered important to include widely used solutions as these can influence user expectations for similar systems.

Literature research: The results of this informal analysis provided the foundation for the next step: the literature research. This research was used to validate or refute the results of the informal analysis. Additionally it also extensively complemented these results by drawing upon more than 30 years of experiences in the areas of CSCW and groupware. As outlined in the historic overview in chapter 2 much of the early research in this field was focused on understanding the specific usage contexts of collaborative work-environments. More recently this aspect seems to be receiving more attention again as connected devices increasingly permeate the environment and enable collaboration outside the confines of traditional - and well researched - work environments. As such some of the early research work and methods were also found to be relevant to this thesis. The literature research combined with the initial analysis resulted in the first iteration for the classification presented in chapter 4. Moreover it also provided the background material for chapter 2.

Classification: Classification is widely used in the sciences to organize populations - originally biological organisms - into different groups and sub-groups according to similarities and differences between them. Within the CSCW and groupware research fields classifications and taxonomies are most often applied to organize individual collaboration tools or more complex collaboration systems according to their usage contexts and application types. A good overview of this type of existing classifications can be found in [Penichet u. a., 2007].

The reason why this new classification scheme presented in chapter 4 was deemed necessary is that the overarching goal of this research effort is to provide an overview of user interface requirements in collaboration systems. The classification therefore provides a foundation for the analysis and comparison of existing solutions in this space, an example of which was documented in chapter 5.

Additionally it also supports software engineers and interaction designers who are working on the design and implementation of new text-based collaboration systems.

As mentioned above other classifications in this area are more focused on broadly categorizing and comparing different types of collaboration systems. As a result the classification presented in this paper was carried out with two major goals in mind:

- **Communication:** The resulting classification scheme is intended to provide a common vocabulary for certain aspects of text-based collaboration system regardless of the specific application context. This should help software engineers and others to quickly grasp the different components of text-based collaboration systems when developing new solutions or analyzing existing ones. The classification scheme can therefore be seen as a supportive tool which is particularly useful during the requirements specification phase of a software project. Additionally the classification scheme also provided a basis for allowing comparisons during the review of different products and services later in this research effort.

- **Reflection:** The underlying process during classification is in-depth analysis of a subject matter and identification of common and differing characteristics which are subsequently used to define the classification scheme. These steps require and support a reflection of complex subject matter, especially when applied in an iterative fashion.

Based on these foundations the classification scheme presented in chapter 4 was created by the aforementioned combination of the results of the initial analysis with the literature research and improving it in an iterative fashion. In essence, the overall approach and goal of this process are similar to the ones described in a recent paper on design space analysis [Hincapié-Ramos u. a., 2011]. However whereas that paper focused on availability-sharing systems and an associated design space this thesis provides a design space which is applicable to all kinds of collaborative systems.

More specifically the following steps were used to develop this classification scheme:

- An initial set of classification criteria was developed by identifying common and differing characteristics of a small selection of text-based collaboration systems.

- These criteria were iteratively expanded and refined by including more products and services during the initial analysis phase.

- Based on the results of the subsequent literature research the criteria were clustered into three different groups, each representing a different aspect of text-based collaboration systems.

- The resulting classification scheme provided the basis for the reviews of products and services in chapter 5.

- Finally the experiences from the review phase were used for improving the original classification scheme.

Reviews: As mentioned above the original classification scheme was used to formally review products and services as presented in chapter 5. Some of them had already been used in the initial analysis while others were new additions. The reviews were conducted with a mixture of exploratory usage combined with studying available documentation such as integrated help and online resources to distill common use cases for the specific product or service.

The results of this review process were compiled into the overview provided in chapter 6. It includes a table mapping the identified design dimensions to their implemention in a small selection of text-based collaboration systems. Additionally it presents a list of good practices and recommendations to be considered when designing or implementing text-based collaboration systems.

Finally the conclusions of the entire research effort were written up in chapter 7.

4 Classification

This chapter details the classification system used in this thesis and more specifically the design dimensions identified during the research. For completeness an overview of implementation criteria is also provided though these are not included in the classification system itself as they lie beyond its scope.

As previously mentioned in chapter 3 the classification was developed to be used for the analysis of existing collaboration systems as well as providing a design space for new software developments. It therefore goes beyond being a supportive scaffolding for the reviews in chapter 5 and represents a core outcome of the research effort documented in this thesis.

4.1 Usage Criteria

As outlined in chapter 2 the nature of collaboration in text-based systems has changed significantly over the past few years. The widespread use of connected and mobile devices such as phones, e-book readers, and tablets will continue to impact the developments in this area. Especially the fact that mobile phones are expected to "overtake PCs as the most common Web access device worldwide"[7] in 2013 will undoubtedly lead to changes in how collaborative products and services are used.

Accordingly the requirements and expectations of users when it comes to using text-based systems in a collaboration context are also changing. At the same time it is important to emphasize that these changes tend to be evolutionary rather than revolutionary with newer developments in areas such as mobile computing slowly entering other usage scenarios.

One crucial aspect which is unlikely to change in the foreseeable future is that there are two key dimensions when it comes to the usage context of collaboration systems: time and space. This classification was first used within the context of groupware research[Johansen, 1988] and has subsequently been adopted and adapted in many other research efforts.

4.1.1 Time

The time dimension is split into non real time and real time. In some services and products these usage scenarios do overlap yet in most cases it is fairly easy to put them in either of these two categories. In some contexts collaboration systems aimed at non real time and real time usage are also referred to as synchronous or asynchronous systems.

- **Non Real Time:** As the name implies the general expectation here is that collaboration does not happen in real time. More specifically this usage scenario suggests that not more than one user is manipulating a digital artifact at the same time.

[7]Gartner: Gartner Highlights Key Predictions for IT Organizations and Users in 2010 and Beyond. `http://www.gartner.com/it/page.jsp?id=1278413`, 13.10.2010 (Last access: 26.07.2011)

An example here is the traditional workflow found in the publishing industry. It generally relies on a strong division of tasks with authors writing the text, editors reviewing it, others working on verification tasks such as spell checking and fact checking, and designers preparing the visual presentation of the content. Beyond the mere division of tasks the important factor in this time context is that the different steps or rather their interaction with a single digital artifact generally happen in a sequential, rather than parallel, fashion. However by splitting up an artifact into independent sub-parts it is possible for different users to interact with and modify it though this introduces additional complexities into the system.

- **Real Time:** The opposite end of the spectrum is real time collaboration whereby multiple users manipulate a digital artifact simultaneously. One example here is the collaborative documentation of conference calls with services such as Etherpad or Google Docs.

  One feature which often indicates whether a service or product is catered towards real time use is the inclusion of presence information such as an availability status or a list of users currently working on a given document. Particularly instant messaging and other communication focused products such as GTalk and Skype make use of such presence information. However increasingly also document focused services such as Etherpad or Google Docs provide similar features.

4.1.2 Space

The space dimension includes same space and different space. Again there are services and products which are designed for both of these usage scenarios however most of them are focused on one of them.

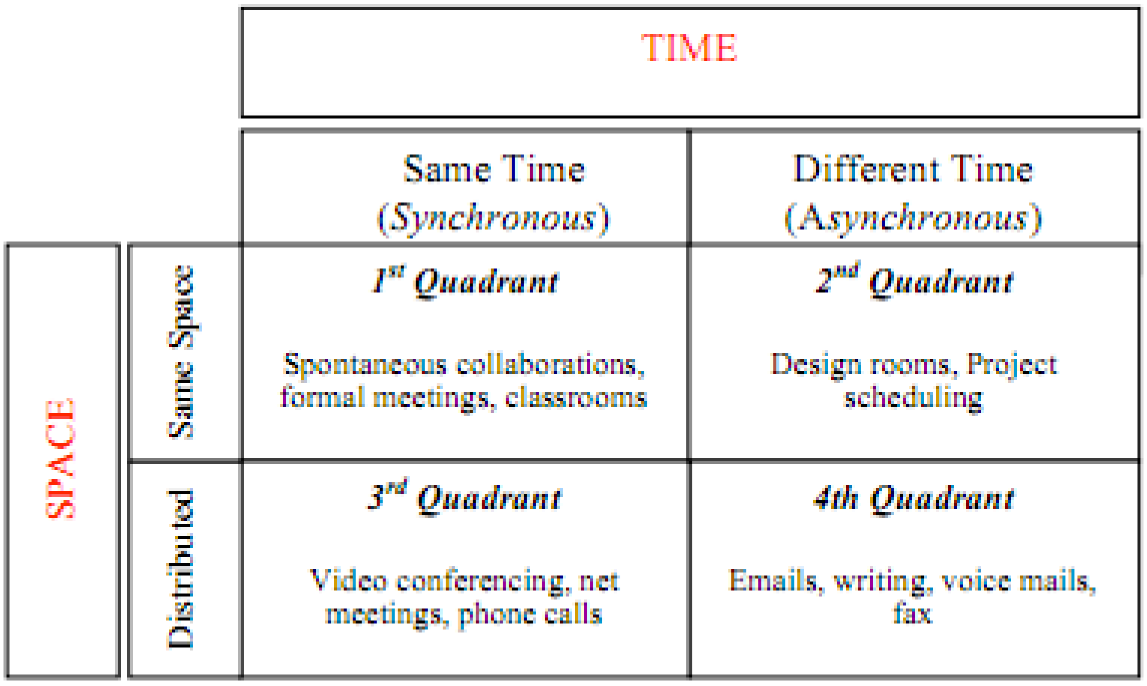
- **Same Space:** The underlying assumption here is that the users of a collaboration system also share a physical space while they are using the system.

  One such example is a project called Slidecasting 2.0 which is being developed at Vienna University of Technology. The system enables students to make live annotations of a lecturer's slides to add questions, comments, or references with additional information. If a video stream of a lecture were provided online then remote users would also be able to use the system yet in its current form it was intended to increase participation within the class.

- **Different Space:** The opposite assumption says that users are in different physical spaces while engaging with the collaboration system.

  There are many examples of services and products, particularly Web based ones, which are designed to support remote collaboration. One of the more widely used ones is MediaWiki which is the software developed to power the Wikipedia encyclopedia. The underlying idea of the Wikipedia project - which still largely shapes the development of MediaWiki today - was to enable large numbers of distributed volunteers to contribute to a common body of knowledge.

The combination of these two dimensions resulting in four different possible combinations is sometimes referred to as CSCW quadrant. Figure 1 is an example of this quadrant provided in a 2006 paper[Rama und Bishop, 2006].



		TIME	
		Same Time (*Synchronous*)	Different Time (*Asynchronous*)
SPACE	Same Space	***1st Quadrant*** Spontaneous collaborations, formal meetings, classrooms	***2nd Quadrant*** Design rooms, Project scheduling
	Distributed	***3rd Quadrant*** Video conferencing, net meetings, phone calls	***4th Quadrant*** Emails, writing, voice mails, fax

Figure 1: CSCW quadrant.

Beyond these two usage criteria there are several important design dimensions which generally need to be considered when analyzing or implementing collaboration systems.

4.1.3 Awareness

One of the most fundamental aspects of text-based and other collaboration systems is awareness. There is an extensive body of research into awareness in the broader area of "Computer-Supported Cooperative Work"(CSCW)Schmidt [2002] and subsequently there are also varying definitions of the term. One of the most widely used ones is "an understanding of the activities of others, which provides a context for your own activity"[Dourish und Bellotti, 1992].

In its most basic form awareness is focused on the presence of other users. [Rittenbruch und Mcewan, 2009] comment that already very early implementations such as the 'who' command - which provides information on other users who are logged onto the same UNIX terminal - could be considered as providing basic awareness information.

In modern real time collaboration systems such as instant messaging applications presence is often communicated with an *availability status*. This indicates whether a user is online in the system though today most services provide more granular options such as Skype's being online but 'away' or 'do not disturb'. In non real time systems presence is communicated with different mechanisms such as for example the locking of files in distributed source code control system - which is explained in more detail in a later section.

However even in systems which do not include an explicit availability mechanism users often communicate their presence via other means. For example in the largely text based Internet Relay Chat (IRC) systems presence is sometimes communicated by users via nicknames. The most popular such use is appending the characters 'afk' to ones nickname. This abbreviation stands for 'away from keyboard' and indicates that the user is currently not seated in front of the computer. This exemplifies the importance of presence information for text-based collaboration systems.

As previously mentioned presence is only a part of the context providing awareness information which are provided in collaboration systems. *Other users' current activity or focus of interest* goes beyond just providing presence information. In communication focused systems such as instant messaging services (e.g. Skype) one widely used additional mechanism is to inform other users once someone has started typing a reply. In other systems such as some real time collaborative text-editors, spreadsheet-editors, or form-based applications it is possible to see which parts of a text or table other users have selected or is in-focus on their screens.

A different level of awareness is provided by *complementary communication tools* which work alongside collaborative systems which are not necessarily by themselves focused on communication. These tools, most widely encountered in the form of asynchronous or synchronous chat or comment functionality, enable a meta-communication without directly affecting the digital artifacts being collaboratively worked on.

4.1.4 Granularity

Within computing the word granularity can have a whole range of meanings. Within the context of collaboration in text-based systems in this thesis it is defined as the units of content which users can operate on.

These range from the smallest possible units such as *individual characters* over units such as *words or individual cells* to *compositions* such as sections, rows, columns or selections and finally up to *entire files, forms, or documents.*

4.1.5 Locking

One aspect which can be found in many collaboration system is the notion of locking. More specifically this refers to a functionality which communicates that a digital artifact, or a granular unit of it, is in the process of being modified by a user, or a group of users. As a result other users are notified when attempting to modify that artifact and depending on the implementation they might also not be able to modify the artifact themselves while the lock is in place.

The latter approach is often referred to as *pessimistic concurrency control*. Especially in database systems there is also the notion of *optimistic concurrency control* which one early article describes as being "based on the idea of conflicts and transaction restart"[Menascé und Nakanishi, 1982] and requires a merging component. Another aspect is that a collaboration system can support *explicit locking*, for example by running a command, or *implicit locking*, for instance by setting the focus on a certain UI element.

Version control systems such as Subversion are an example of a widely used collaboration system which makes use of locking. Beyond the mere locking described earlier Subversion also offers options such as allowing users to *remove or steal locks* from others. Administrators of the systems are also able to remove these locks and in some systems also support options such as locks having a specified timeout after which the artifacts are again available to everyone.

4.1.6 Merge Transparency and Forking

Another aspect of collaboration systems which is closely related to locking and concurrency control is merge transparency and branching/forking. One possibility is for a collaboration systems to hide the complexity of allowing different users to independently work on a single digital artifact with *transparent merging*. However in some cases it is necessary or desirable to *expose merging mechanisms* by making them explicitly available or visible to users.

When the copy of a text file has been independently modified by different user and upon trying to merge them into a single output contradicting information is encountered a decision on which information will be accepted is required. In some systems such a decision can be configured to be taken automatically, for example by generally discarding older changes, and the result visualized for the users. Another possibility is to show the contradicting parts of an artifact to users and allowing them to decide.

Similarly many collaboration systems such as source code management solutions also offer functionality for the explicit creation of new branches or forks. This enables a temporary or possibly even permanent independent modification of artifacts. In software development branches are commonly used to distinguish between versions focused on bug fixes and ones including new functionality. Forking is a practice which is commonly encountered in open source software development and is associated with starting independent developments based on a common platform.

4.1.7 Mobility

As mentioned earlier the rapidly increasing importance of mobile devices also has an impact on text-based collaboration systems. Mobility and the use of mobile devices generally require a different approach to product and services design. One publication editorial[Dunlop und Brewster, 2002] provides a list of five fundamental design challenges specific to the use of mobile devices:

- Designing for mobility
- Designing for a widespread population
- Designing for limited input/output facilities
- Designing for (incomplete and varying) context information
- Designing for users multitasking at levels unfamiliar to most desktop users

The rapid development of mobile devices has in many ways brought them closer to traditional computer systems in terms of their technical capabilities. However these challenges still need to be addressed in modern mobile products and services. *Adaptive user interfaces*, *limited functionality designs* such as read-only representations, *support for offline use* and *designing for interruptions* are such possibilities to address the specific challenges of collaboration on mobile devices.

One consideration which is not explicitly included in the aforementioned list is that today many products and services are offered for both mobile and non-mobile environments. For example the popular Evernote note-taking platform, which also supports sharing and collaboration features, is available as a browser-based Web service, native OS X and Windows applications, and apps for a broad variety of mobile platforms such as Android, BlackBerry, iOS, and others.

This example indicates that text-based collaboration systems could become more heterogeneous in the future. As such the underlying systems and technologies have to be designed with different access mechanisms in mind. Additionally users on different access devices might have different requirements and expectations with regard to their experiences and manipulation capabilities. In a collaboration environment multiple users might also be accessing the same digital artifact by using different access mechanisms.

Moreover not only will different users rely a variety of access mechanisms but within a collaborative session individual users might switch between them. Joshua Topolsky, a technology blogger, has coined the term “continuous clientto describe such a system which supports that “when you leave one device, you pick up your session in exactly the same place on the next device you use”[8]. Likely the most widely used example of such a system is Kindle Whispersync by Amazon which is similarly described as a technology that “saves and synchronizes your last page read, bookmarks, notes, and highlights across your devices.”[9] *Support for seamless switching of devices* enables such continuous collaboration across mobile and desk-bound use situations.

[8]Topolsky, Joshua: A modest proposal: the Continuous Client. `http://www.engadget.com/2010/05/26/a-modest-proposal-the-continuous-client/`, 26.05.2010 (Last access: 17.08.2011)

[9]Free Kindle Reading Apps. `http://www.amazon.com/gp/feature.html/`. (Last access: 17.08.2011)

4.1.8 Ownership and Security

Another important component of collaborative systems is ownership. As with many other concepts detailed in this chapter the notion of ownership is also very dependent on the specific application context. Generally speaking however the key issue is whether a user, or also group of users, is given additional possibilities and/or responsibilities when it comes to managing a collaborative session. In a workflow driven system this could for example mean additional functionality to sign off on or revert revisions of a document. In a communication product the addition and removal of people to a multi-user conversation is often handled by a single user.

One aspect of ownership which can be found in many collaborative text-based systems regardless of the specific context is that default ownership is based on who initiated a collaborative application or service.

The related notion of having different *levels of ownership* can also be found in many systems. For instance Google Docs offers the following four levels of access to documents and folders[10]:

- **Viewer:** Is limited to viewing and exporting documents and the contents of folders.
- **Commenter:** Besides the viewer's permissions a commenter can also leave comments on documents.
- **Editor:** Can additionally modify documents, move the contents of folders, view the list of editors, and - if given the permissions - invite other editors and viewers.
- **Owner:** Can in addition always add more editors and viewers as well as delete documents and folders.

What is important to note is that Google Docs only allows a single user to be the owner of a document at any given time.

This is also touches upon the related aspect of *transfer of ownership*. In the aforementioned Google Docs an owner can invite other users as owner, editor, or viewer of document. Another approach is taken by many installations of the popular open source solution EtherPad where knowing the URL of the document is enough to gain full read and write access. As a document's URL is randomly assigned the user starting it has to share it with collaborators however afterwards he has no further control over access to it. Some newer installations of EtherPad do make a distinction between users who can only view a document and those with edit permissions. However again the initiator of a document looses his implicit ownership as soon as he shares the document's link with others.

Ownership is closely related to the wider notion of security which is a major concern when developing software systems, even more so in connected environments. Since collaboration systems necessarily

[10]About owners, editors, and viewers. `https://docs.google.com/support/bin/answer.py?answer=50085` (Last access: 13.09.2011)

require different users to access common digital artifacts, security within them requires additional components and dimensions. One paper refers to these additional dimensions as “application level as well as organizational level attributes”[Ahmed und Tripathi, 2010]. The role of the user within the collaboration context and the previously discussed ownership levels are examples of such attributes.

Of particular concern within collaboration systems is the *confidentiality* aspect of security. This aspect is generally defined as preventing users from accessing information they do not have permission to access. In collaboration systems this does not only concern the digital artifacts themselves but also related metadata such as when changes were made, who made them, or simply who even has access to an artifact. A related consideration is the possibility to remove such metadata entirely before passing the original artifact on to other users or publishing it.

It is also noteworthy that security within collaboration systems may conflict with requirements in other categories such as awareness.

4.1.9 Scalability

The term scalability is commonly associated with the technical implementation as to how many users can concurrently access a Web service. However within the context of text-based collaboration systems there are additional usability aspects which need to be taken into consideration. It may for example be technically feasibly to allow 100 users to place locks on parts of a document or form. Yet from a user’s point of view such a functionality would likely lead to frustration as keeping track of when one can access previously locked resources would be quite time-consuming. Scalability with regard to usability is therefore focused on design decisions such as how many and how users can concurrently or sequentially collaborate.

4.1.10 Version Tracking

Keeping track of changes which were made between different versions of a single digital artifact is another usage criteria of collaborative systems. In many systems this tracking of changes happens on two different levels.

First of all systems save the *actual differences* made between versions in order to later present or visualize them for the users. Secondly users can often also *attach metadata* such as comments to changes. This type of explicit documentation is a way to make it easier for others to comprehend or follow such changes.

Another functionality which is sometimes available in the context of version tracking is a *notification system* which informs users about changes made to digital artifacts. The owner of a Google Docs spreadsheet for example can define when to to receive a notification e-mail by selecting from the following options:

- Any changes are made

- Anything on this sheet is changed
- Any of these cells are changed
- Collaborators are added or removed
- A user submits a form

Another benefit of version tracking is that it allows for *revision control* of digital artifacts. Within the context of collaborative systems and particularly in combination with the aforementioned ownership criteria this is a versatile tool for controlling, reviewing, and quickly undoing changes. Such functionality is available in a wide range of text-based collaboration systems from communication tools and document focused products to source code management services.

As many other usage criteria version tracking is also closely associated with the awareness criteria introduced earlier in this chapter.

4.1.11 Visualizations

The visualization of data and information is an aspect which generally goes beyond regular usability considerations. However given the complex relations and processes which are often represented in collaboration system visualizations are an important component of making such systems more usable.

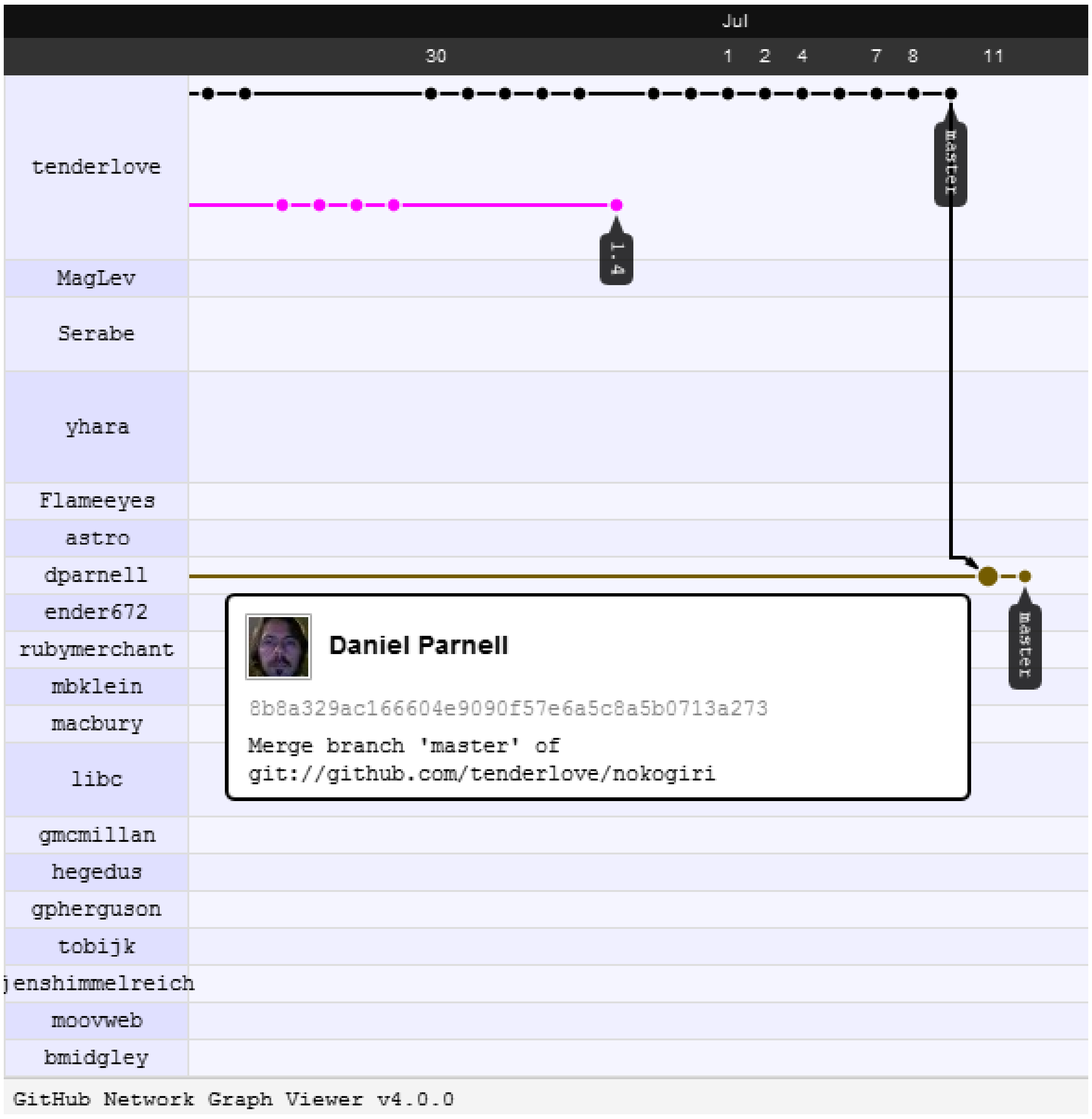
One example of such a visualization is the Network Graph Viewer introduced by the source code hosting service GitHub in late 2008[11]. It visualizes the commits, branches, and merges of users and repositories connected to each other. All of this information is collected as part of the version tracking discussed earlier and is also available textually in the history list of commit, fork and merge actions. Yet it is the visualization which connects the dots and thereby makes the information easily accessible to the users. Please see Figure 2 for a screenshot of GitHub's Network Graph Viewer.

4.2 Implementation Criteria

In chapter 5 a review of several different products and services in the area of collaboration in text-based systems will be conducted. That review will focus on the usage criteria outlined above. However as the goal of this thesis is to also support the development of collaboration systems the following software implementation characteristics which are highly relevant to such systems were included.

- **Extensibility:** This characteristic emphasizes the possibilities of extending the product or service beyond its original capabilities. This can be achieved via components such as plug-in systems or public APIs. This point must not be confused with openness mentioned below as it is possible for systems to be extensible without necessarily being very open.

[11]Preston-Werner, Tom: Say hello to the Network Graph Visualizer. `https://github.com/blog/39-say-hello-to-the-network-graph-visualizer`, 10.04.2008 (Last access: 10.07.2011)



Daniel Parnell

8b8a329ac166604e9090f57e6a5c8a5b0713a273

Merge branch 'master' of
git://github.com/tenderlove/nokogiri

GitHub Network Graph Viewer v4.0.0

Figure 2: GitHub Network Graph Viewer of a software project.

- **Flexibility:** At first sight flexibility might appear to be very similar to aforementioned extensibility. Though whereas extensibility focuses more on the technical implementation flexibility is a broader quality. It can encompass aspects such as whether a particular solution is applicable in a broad set of environments as opposed to being very domain-specific or if it can be used equally well on mobile devices such as mobile phones and traditional computers.
- **Openness:** This dimension was added due to practical considerations. First of all approaches which are publicly documented and products or technologies which are free and open source software allow for future software projects to explicitly learn from prior experiences and (re-use) available building blocks. Secondly openness also includes the possibilities of users to move data out of a product or service and potentially into other systems.
- **Reliability:** This dimension deals with the behavior of software under extraordinary circumstances such as a crash of the system it is running on. Particularly the resilience against data loss is an important consideration here.
- **Scalability:** Especially with many Web services scalability in the sense of how many users or parallel accesses a system can deal with is an important issue. It also needs to be considered separately from reliability. At the same time scalability also extends beyond characteristics such as response times into areas such as the user interface and general usability.
- **Security:** Security is an important consideration in any software project and its importance only increases in a collaborative environment with multiple users and different levels of access privileges. Apart from ensuring that users only have appropriate access additional considerations such as whether metadata associated with documents can be removed before publication have to be taken into account.
- **Software Architecture:** While all of these implementation criteria are part of the larger software architecture this point specifically refers to high-level decisions such as whether to use a centralized or decentralized software system. Such key implementation aspects can have a significant impact on design dimensions of collaboration systems.

4.3 Discussion

The classification presented in this chapter attempts to be a comprehensive resource to support the analysis of existing as well as development of new collaboration systems. Given this focus there are however some inherent limitations.

For example it could be argued that the software architecture should be part of the design space. After all decisions such as whether to build a centralized or decentralized collaboration system can have impacts on several other characteristics in the design space. However it was decided that the software architecture is more closely related to the implementation rather than the design stage and therefore it was added to the list of implementation criteria. Following up on how these implementation details impact the design space is beyond the scope of this thesis, though it might be worth following up such considerations in future research.

4.4 Outlook

Given current developments it can be assumed that in the future more technologies, services, and products will adopt a hybrid approach enabling collaboration across non real time and real time as well as same space and different space environments.

One example of a significant move into this direction are the updates to its messaging system which Facebook started introducing in late 2010[12]. Of particular interest in the time and space dimensions is that regardless if they were sent in a synchronous or asynchronous scenario or via chat, text message, or e-mail from any number of devices all Facebook messages are accessible in a single place. This makes it easier for users to continue a thread of conversation regardless of which context previous messages were written in.

Such developments will continually impact the relative importance of the various usage criteria mentioned earlier. Especially mobility related considerations and designing around the affordances and constraints imposed by the different categories of mobile devices will become more relevant. This is due to the fact that people are increasingly using mobile devices as their primary means of connecting to networks and collaboration systems. When increasing amounts of work is done in such systems the importance of awareness of other users' actions is also likely to increase. Similarly the additional data provided by the context of mobile users and increasing amounts of information due to more extensive use will lead to higher demands when it comes to the visualization of such data and associated metadata.

At the same time the fundamental usage dimensions and implementation criteria described in this thesis are likely to remain constant for the foreseeable future. However as mentioned above it can be expected that an increasing number of solutions will try to address more than one of the requirement profile combinations embodied in the CSCW quadrant. As outlined above this change is largely due to the increasing ubiquity of mobile and connected devices which allow users to access collaboration systems regardless of their physical location. This is contrary to the past where traditionally users had to sit at fixed locations in other to be able to use such systems.

With regard to the implementation criteria the expected ubiquity of access to collaboration systems could impact the relative importance of scalability as increasing numbers of users will want to use these services and products. In parallel security considerations are likely to become even more relevant as an effect of the broader use of collaboration systems, particularly when it comes to sensitive domains such

[12]Seligstein, Joel: See the Messages that Matter. `http://blog.facebook.com/blog.php?post=452288242130`, 15.11.2010 (Last access: 10.05.2011)

as medical systems. A related consideration is the importance of graceful degradation of collaboration services in situations where connectivity is not available.

4.5 Summary

The purpose of this chapter was to describe the classification system used for the reviews documented in chapter 5. The division into usage criteria and implementation criteria is based on the two core areas of collaboration systems. Hereby the usage criteria outline the usability considerations and functionality which such systems need to address in order to allow for effective and efficient use. The implementation criteria then provide a set of guidelines regarding the dimensions in the software implementation which need to be considered in collaboration systems.

The goal of this set of descriptions is to facilitate the process of observing and understanding differences and similarities between technologies, products, and services in the area of text-based collaboration systems. It thereby also enables focused design considerations for all of these criteria.

As a specific example this classification system was already used during the requirements engineering stage of a prototype for a collaborative form-based system for documentation of clinical studies.

5 Review of Products and Services

This chapter contains reviews of selected products and services which offer text-based collaboration.

The goal of the selection process for the products and services presented in this chapter was to include a small selection of both

- Those products which have proven to be popular with users and have received extensive user exposure thereby also influencing expectations for similar systems, as well as
- those using particularly innovative building blocks in terms of technical implementation, usability, and/or application context

It is also important to emphasize that due to the many different contexts and environments for which these products and services were and are being developed for it is neither feasible nor very useful to draw direct comparisons between them.

As previously mentioned the review of products and services focuses on the usage criteria introduced in chapter 4. The implementation criteria also outlined there are not part of this review as they lie outside the main scope of this thesis.

5.1 Etherpad

Etherpad is a collaborative document editor which was released as open-source software after Google acquired the original development team in late 2009. Today work on the platform is coordinated by the Etherpad Foundation and there are a number of public sites which offer Etherpad or Etherpad-based solutions to users.

- **Time:** Etherpad is described as having been the first real time collaborative editor on the Web when the project was launched in late 2008. At the time only native desktop applications had been able to provide real time performance. So while Etherpad's technical contributions were focused on providing real time editing capabilities the software can also be used for non real time collaboration.
- **Space:** In most cases document focused collaboration systems such as Etherpad can be used in same space and different space configurations. Yet their main focus it set on enabling communication and collaboration across different spaces, be it within one office or across continents.
- **Awareness:** Etherpad uses *visual cues provided in real time* to display awareness information. Basic presence information in Etherpad is available via a *list of users* who have opened the same document. Below that list of currently connected users it also offer a *chat window* for additional communication. An noteworthy aspect here is that in Etherpad the chat history is permanently saved together with the document. Thereby the chat not only facilitates real time but also non

real time collaboration. When it comes to showing what other users are currently working on in the document Etherpad assigns a different default colour to every user. Subsequently all text written or edited by that user will be presented with these *'authorship colours'*. While this option can also be turned off most Etherpad installations analyzed for this review stuck with this default behaviour.

- **Granularity:** In terms of granularity Etherpad enables collaboration on an *individual character basis*. Additionally it also employs the concepts of different *documents* which are all uniquely identifiable via their URLs.
- **Locking:** Etherpad *does not support any form of locking* which means that all parts of a document are accessible to every user who knows the URL of the document. There is also no possibility for users to implicitly lock parts of the text by selecting it.
- **Merge transparency and forking:** The way in which Etherpad documents are kept consistent is through *transparent merging* whereby users do not have any explicit capabilities to control or configure the process. Similarly Etherpad *does not offer any fork functionality* within the system. The only feature which could be considered as providing a minimal fork capability is the possibility to link to specific revisions of an Etherpad document or export it to common text formats such as HTML, OpenDocument, PDF, and Word. This enables the preservation of a specific state of the document.
- **Mobility:** No specific mobility-focused versions of Etherpad are available at this point.
- **Ownership and Security:** Etherpad uses a fairly *simple ownership model*. Everyone who has access to an Etherpad document via its URL can be considered an owner of the document as he can edit anything within the document, use the chat functionality, and in some versions of Etherpad even modify the general viewing options for that specific document. The only exception to this model is a special *read-only link* which can be passed on to other users who will only be able to see the document or a specific revision of it.
- **Scalability:** When it comes to scalability a post on the Etherpad blog from early 2012 recommends a maximum of 15 concurrent users[13] though it is not clear whether this number is dictated by the technical implementation or other factors. From a usability perspective there are no obvious reasons as to why the number of concurrent users could not be higher. As part of this review the author participated in an Etherpad session with up to 12 concurrent users and aside of slight performance-related issues the collaboration itself did not seem to suffer under the large number of participants.

[13]McLear, John. Etherpad Vs Etherpad Lite - Which is right for you?. `http://etherpad.org/2012/01/15/etherpad-vs-etherpad-lite-which-is-right-for-you` 12.01.2012 (Last access: 30.07.2012)

- **Version tracking:** Internally Etherpad uses operational transformation with changesets to track the changes made to a pad [14]. This allows for a very granular version tracking and step-by-step undo and redo functionality. Etherpad also exposes this information via a feature called Time Slider. Essentially Time Slider allows users to replay the entire change history of a pad. This replay is however limited to the text in the pad itself, contextual information such as the chat is not saved. Etherpad also does not provide any facilities to attach metadata to changes and explicitly saved revisions.
- **Visualizations:** Aside of the aforementioned Time Slider Etherpad does not provide any facilities to visualize the information or processes captured in a pad.

5.2 GitHub

GitHub is a Web-based system based on the source code management software git. git was originally designed by Linus Torvalds for development of the Linux kernel. As of mid-2011 GitHub was reported to be the world's most popular online source code management system[15] and in early 2013 the service announced that it had more than three million users and hosted almost five million code repositories [16].

- **Time:** One of the core uses of source code management systems in software development is to enable different people to collaborate. Therefore GitHub supports non real time and real time collaboration whereby different developers can work on the same source code simultaneously. However changes to the source code are not propagated in real time so aspects such as branching and merging- which are discussed below - play an important role.
- **Space:** Similarly to the other products and services included in this review GitHub is equally well suited for same space and different space collaboration.
- **Awareness:** Neither GitHub nor the underlying git system provide *presence* information. However GitHub does include a notification system which provides information about *other users' current activity.* It is possible to watch and be notified of all activities related to selected projects and also to follow individual users to keep track of their contributions across different projects. In that sense GitHub is a good example of a Web service which adds tools to facilitate collaboration to an underlying software solution.

[14] AppJet, Inc. and Etherpad Foundation. Etherpad and EasySync Technical Manual. `https://github.com/Pita/etherpad-lite/raw/master/doc/easysync/easysync-full-description.pdf` 26.03.2011 (Last access: 30.08.2012)
[15] Finley, Klint. Github Has Surpassed Sourceforge and Google Code in Popularity. `http://readwrite.com/2011/06/02/github-has-passed-sourceforge` 02.06.2011 (Last access: 13.03.2013)
[16] Sanheim, Rob. Three Million Users. `https://github.com/blog/1382-three-million-users` 02.06.2011 (Last access: 04.03.2013)

- **Granularity:** GitHub follows in the foot steps of older stand-alone programs such as diff, patch, and merge in that its granularity is also line oriented. When uploading new or changed source code to the platform it counts and visually presents any changes as additions or deletions of lines. The next granular unit are files for which all modifications are kept tracked of. Last but not least changes which are uploaded to the system are collected as so-called commits which can be reviewed and reverted as individual granular units.

- **Locking:** Contrary to the older Subversion source code management system git, and therefore also GitHub, take a distributed approach which does not support locking of files. Rather it aims to avoids the issues which locking mechanisms usually try to address in other systems by different approaches such as the facilitation of local cloning of repositories and subsequent merging of different versions.

- **Merge transparency and forking:** As git and GitHub is particularly catered towards use in open source software projects which encourage people to contribute the creation and management of branches and forks and their subsequent merging is an important functionality. Hence *transparent merging* is supported but explicit *merging mechanisms* are also exposed.

- **Mobility:** GitHub provides official Android and iOS apps. However these apps focus on keeping track of issues and news and discussing short code pieces (Android-only) and don't provide any collaboration features making comments.

- **Ownership and Security:** With regard to ownership GitHub has different *levels of ownership* depending on whether the user in question is working with a user account or an organization account. User accounts have two levels: owner and collaborator. Organization accounts provide more granularity by supporting owners, administrative teams, and so-called push teams with decreasing levels of rights and capabilities. For user accounts *transfer of ownership* isn't possible and other users can only be granted collaborator access. Organization accounts however do support the full *transfer of ownership*.

- **Scalability:** Any GitHub project allows for an unlimited number of collaborators. In popular projects with many contributors keeping track of all changes and the associated notifications can be challenging from an organizational point of view. However the software itself does not exhibit any limitations in terms of the collaboration capabilities themselves.

- **Version tracking:** Version tracking is one of the core features of any source code management system so it comes as no surprise that GitHub provides a broad set of functionality in this area. Besides the *actual differences* GitHub also requires users to *attach metadata* in the form of commit messages to each change which is uploaded to the system. As mentioned in the awareness section above it also provides a *notification system* to users. Last but not least GitHub offers powerful possibilities for *revision control* whereby it focuses on two popular models of managing changes. Fork & pull is catered towards open projects with many contributors whereas shared repositories are more commonly used in private projects with a smaller number of developers[17].

- **Visualizations:** GitHub was already mentioned as an example of a collaboration service which provides visualizations in chapter 4. However it doesn't just offer that aforementioned Network Graph Viewer but also other visualizations which show code contributions over time, commit activity over the last year, an overview of code line additions and deletions, and at what times and days commits were made to a project [18]. This data is all implicitly collected by the system and also available in other places. However by making the data more accessible the visualizations provide additional value to the users.

5.3 Google Docs

Google Docs is a suite of collaborative office productivity tools which is offered online. These tools includes a word processor, a spreadsheet processor, a presentation software, a collaborative drawing tool, and a collaborative form editor. Within the scope of this thesis the focus of the review were the word and spreadsheet processor.

- **Time:** Similarly to Etherpad one of Google Docs' strengths is its support of real time collaboration and its real time chat functionality also supports such a usage scenario. However at the same time it provides a comment feature which facilitate non real time use. So overall Google Docs covers the complete spectrum of real time, non real time, and mixed time use.

- **Space:** With regard to this criteria there are again parallels between Google Docs and Etherpad. Google Docs is equally suitable for collaboration within the same space as it is for working across different countries and continents.

- **Awareness:** Google Docs uses *visual cues provided in real time* to display awareness information. Basic presence information in Google Docs is provided via a *status message* saying "x other viewers". This message can be clicked to show a list of these other users. Below that list of currently connected users there is a group chat window for additional communication. These chat messages are not stored in-between sessions. However for more permanent and asynchronous meta-communication Google Docs offers a facility to leave comments on selected parts of a document or spreadsheet. Additionally it also allows users to configure if and when they want to receive notifications of such comments. When it comes to written text Google Docs does not utilize 'authorship colors'. Rather it shows colored cursors at the locations where other users' cursors are currently positioned. Additionally it also shows text areas others have highlighted.

[17]GitHub Help. Using Pull Requests. `https://help.github.com/articles/using-pull-requests` (Last access: 04.03.2013)

[18]GitHub Help. Using graphs. `https://help.github.com/articles/using-graphs` (Last access: 04.03.2013)

- **Granularity:** Like most other modern text-based collaboration systems Google Docs' word processor also allows for collaboration on an *individual character basis*. Similarly the smallest granular units in the spreadsheet processor are individual cells. The next larger granular unit are individual documents or spreadsheets. These documents are used for access control and similar functionality.

- **Locking:** The word processor does not support any form of locking on parts of a document. Even parts of a text which are highlighted by other users can be modified. The spreadsheet editor offers more functionality in that regard even though it is also possible to edit cells which are currently being edited by others. It is however possible to lock ranges of cells and columns as well as individual sheets by restricting access to them to certain users. In some sense this functionality pertains more to the ownership dimension discussed below. On a practical level it can also be used to temporarily ensure that others are not editing certain cells.

- **Merge transparency and forking:** In Google Docs documents and spreadsheets are kept consistent through *transparent merging*. Thereby users do not have any explicit capabilities to control or configure the process. With regard to forking it is possible to make a copy of a document or spreadsheet. Beyond the document itself this also allows users to copy the collaborators to the new instance. The comments and revision history of the original document are however not available in the copy.

- **Mobility:** There are two ways to work with Google Docs on mobile devices. For Android and iOS devices specific apps are available. Additionally the service can also be accessed via a browser on mobile devices. At time of writing there were significant differences between these access methods. The most comfortable and versatile possibility is using the Android app as it provides a similar range of functionality as the online version. It also comes with *support for offline use*. The iOS app is more limited in its feature set, especially since it does not offer an in-app possibility to edit documents or spreadsheets. Access via a mobile browser is more catered towards viewing documents rather than editing them as for example it is not possible to format text.

- **Ownership and Security:** As already described in chapter 4 Google Docs uses what can be described as a fairly standard model of ownership. *Default ownership* is assigned to the user who initially creates a text or spreadsheet document. In terms of *levels of ownership* Google Docs differentiates between viewers, editors, and the owner. The owner can invite other users to become editors and viewers, including the possibility to give editors permission to invite other users. One particularly interesting aspect with regard to this *transfer of ownership* is that only a single user can be the owner of a text or spreadsheet document at any given time. When it comes to security and especially *confidentiality* Google Docs does not provide any explicit functionality to remove information such as metadata from a document. This metadata, especially the revision history, is only removed upon exporting a document and making a copy of it within Google Docs. The latter option can be considered an alternative though it does result in some overhead. As for viewing metadata it is important to point out that only editors and the owner of a document can see the list of collaborators, viewers and commenters do not have access to that information.

- **Scalability:** There are limits to the number of people who can collaborate on Google Docs documents. More specifically any single document can be shared with a maximum of 200 individual users, unless the document is made public that is. In terms of concurrent collaboration the maximum is 50 users who can view or edit a document at the same time. [19]

- **Version tracking:** Just like Etherpad Google Docs also uses operational transformation to track and process the changes made to documents and spreadsheets. An area where these two services differ is how these changes are presented to the user and stored. First of all per default Google Docs groups individual changes and revisions together "into short time periods to make it easier for the user to identify the slight differences between previous document versions"[20]. But it is also possible to access a more fine-grained history. Secondly Google also uses what it calls revision pruning to automatically limit the amount of revision data which has to be stored.[21] As already discussed in the definition of version tracking in chapter 4 the Google Docs spreadsheet processor also enables users to set up and configure notifications when changes are made to a spreadsheet.

- **Visualizations:** Google Docs does not come with any features to visualize the information in or processes which occurred during the development of a document.

5.4 MediaWiki

Wiki systems have become increasingly popular since the idea of enabling quick and collaborative editing of Web pages was first introduced by Ward Cunningham's WikiWikiWeb project in 1995. However no other wiki has received as much attention as Wikipedia which today contains an estimated 20 million articles in more than 250 languages. As such MediaWiki, the software platform powering Wikipedia, also is the most widely used wiki solution and was therefore included in this review.

- **Time:** Even though it does provide some support for synchronous use MediaWiki is catered towards non real time usage scenarios. In particular it's lack of real time propagation of changes and awareness information (as detailed below) are significant limitations when it comes to synchronous editing of pages.

[19] Google. Limits on sharing. `http://support.google.com/drive/bin/answer.py?hl=en_answer=2494827_rd=1` (Last access: 01.09.2012)

[20] Google. Revision history. `http://support.google.com/drive/bin/answer.py?hl=en&answer=190843` (Last access: 01.09.2012)

[21] Google. Revision pruning. `https://support.google.com/drive/bin/answer.py?hl=en&answer=95902` (Last access: 01.09.2012)

- **Space:** With regard to this criteria MediaWiki is similar to GitHub, Google Docs and Etherpad in that it is equally suitable for collaboration in same space and different space scenarios.
- **Awareness:** In MediaWiki no presence information is available to users. The only exception is when a user tries to save an edited revision of a page if another user edited the same lines of the entry in the meantime. Essentially the system tries to do an automatic merge of the changes and if this is not possible it presents both versions and the differences between them to the last user who tried to save the entry. As such the only way to propagate presence information is by adding explicit notes, such as Wikipedia's 'In Use' template, to entries when editing them.
- **Granularity:** The most important granular unit in MediaWiki is a *page* and most of the functionality is built around individual pages. Internally *lines of text* are an important granular unit for merging and version tracking and therefore also exposed to users at certain times (e.g. when comparing different revisions of a page).
- **Locking:** As indicated in the awareness section MediaWiki employs *optimistic conflict resolution* with a simple form of *implicit locking*. If the system encounters a conflict it cannot solve automatically the last user to have hit the save button will have to do so manually.
- **Merge transparency and forking:** As mentioned earlier MediaWiki *uses transparent merging* when possible and only *exposes merging mechanisms* when contradicting revisions of a document are encountered. Other than that it does not offer possibility for explicitly merging or forking pages.
- **Mobility:** No specific mobility-focused versions of MediaWiki are available at this point.
- **Ownership and Security:** Per default current versions of MediaWiki are configured with seven different groups of users which range from anonymous users to administrative roles such as sysops and bureaucrats[22]. However since there are several dozen individual user rights available in the system the number of roles in specific instances can be more varied. In its default configuration MediaWiki does not consider any users to be owners of a page and access to it is dependent on users' roles rather than who initially created it.
- **Scalability:** Due to its simple approach to locking and merging MediaWiki there are some limitations to its scalability. As such it comes as no surprise that large MediaWiki installations such as Wikipedia make explicit recommendations to its users on how to prevent the resulting conflicts: "One of the easiest ways is to edit the smallest portion of the article necessary."[23]
- **Version tracking:** MediaWiki supports many different aspects of version tracking. Aside of the *actual differences* users can also *attach metadata* such as a comment or indicating that a

[22]MediaWiki. Manual:User rights. `http://www.mediawiki.org/wiki/Manual:User_rights` (Last access: 28.10.2012)
[23]Wikipedia. Help:Edit conflict. `http://en.wikipedia.org/wiki/Help:Edit_conflict` (Last access: 28.10.2012)

new version is only a minor edit which is used to signal small changes related to the spelling or formatting. All of this information is available via a page's history view which offers direct comparison views between different revisions of a page. It is also possible to "watchëntries to keep track of the latest changes to them on a separate overview page. With regard to *revision control* MediaWiki instances such as Wikipedia can choose to protect entries which means that changes need to be reviewed before the new revision is accepted.

- **Visualizations:** The aforementioned comparison view enables a quick check of the differences between two different revisions of an entry.

5.5 Skype

Skype is one of the most popular online communication solutions today, it reported having 663 million registered users at the end of 2010[24]. While Skype is most widely known for its audio and video calling features it also includes instant messaging functionalities and other collaboration methods such as screen sharing. There are three aspects which are particularly noteworthy about Skype: the size of its own user base, the availability on many mobile platforms, and a mid-2011 deal with Facebook. This agreement started an integration process which now allows users to seamlessly chat between the two platforms.

For this review Skype was used on a laptop running Windows 7 (Skype version 5.10.0.114) and on an Android mobile phone (Skype app version 2.5.0.160). The review focused solely on the text-based collaboration aspects represented by Skype's instant messaging capabilities.

- **Time:** As the name implies instant messaging solutions are first and foremost designed to be used for real time communication. However many instant messaging solutions, including Skype, also some support for sending so-called offline messages to users who are not currently online. One interesting design decision here is that Skype only forwards offline messages when both the sending and receiving party are online again. Other instant messaging services store messages on a server and then forward them once the previously offline recipient comes online again.

- **Space:** Like many other collaboration systems Skype can be used in both same space and different space scenarios. Yet its main focus it set on enabling communication and collaboration across different spaces.

- **Awareness:** As with many other instant messaging solutions awareness information in Skype is mostly built around the presence of other users. The main interface for that presence information is the *contacts* list which displays which contacts are online and offline. This basic presence information is enhanced by allowing users to set their *availability status* to 'online', 'away', 'do

[24]Telecom.Paper: Skype grows FY revenues 20%, reaches 663 mln users. `http://www.telecompaper.com/news/skype-grows-fy-revenues-20-reaches-663-mln-users`, 08.03.2011 (Last access: 28.11.2011)

not disturb', 'invisible' or 'offline'. Last but not least it is also possible to set a *status message* which is visible to a user's contacts. Aside from being available within the program or app itself the online status can also be made available to others via badges on Web sites or in e-mail signatures.

When users start typing a reply during instant messaging conversations this information is also broadcast to that users' conversation partner(s) who see a "<username> is typing..notification thereby being made aware of the *other user's current activity*. It is also possible to disable this feature in the program's configuration.

Skype on Windows also provides a variety of other *notification options*, many of which can be configured by users. For example it is possible to receive system tray notifications for a broad variety of events ranging from contacts going online/offline, conversations being initiated or having been initiated while one was offline, or a new user sharing his contact details.

The Skype for Android app provides a very similar set of features and configuration options. One aspect which is less elegantly solved on the Android app than in the Windows program is that when displaying all contacts they are listed in alphabetical order rather than displaying contacts which are online at the top of the list.

- **Granularity:** With regard to granularity Skype is very much focused on *individual messages* and offers manipulation options for them. For example it is possible to edit single messages in an ongoing conversation after they have been sent. Similarly there is an option to quickly copy individual messages from a conversation.

  Additionally Skype also uses the concept of different *conversations* depending on which other contacts are or were involved in it. It is possible for the initiator of a group conversation to add or remove individual users from it.

- **Locking:** Skype is built around the notion of conversations consisting of messages from individuals. These explicitly lead to a common artifact so there is no need for a traditional locking mechanism. As such it is possible for everyone to continually send messages to a conversation which can't be explicitly locked unless a specific user is removed or blocked from a conversation by its initiator.

- **Merge transparency and forking:** Skype does not offer any mechanisms for merging messages or conversations. It is however possible to in some sense fork conversations by adding new contacts to an existing one. Per default the newly added participant will not be able to see the chat history from before having being added. This option can however be changed to make the last 400 messages or two weeks of conversations available to newly added participants.

- **Mobility:** Aside of being available for Linux, Mac OS X, and Windows Skype can also be installed on a large variety of mobile phone platforms such as Android, iOS, Symbian, and Windows

Phone. Last but not least there is also a version for the PS Vita game console. All of these mobile versions are adapted to the specific requirements of each platform and provide a full set of instant messages and voice - and where available - video call features. Skype supports *seamless switching of devices* as any messages composed or received on an account are forwarded to all devices which are currently logged into the system. However some changes are not propagated this way, e.g. setting the availability status on the Windows application automatically changes the availability status on the Android app accordingly but this is not true the other way around.

- **Ownership and Security:** As previously mentioned there are two granular units in Skype: messages and conversations. Each of them has different ownership functionality attached to it. Messages are owned by their creators and can only be edited and deleted by them. For conversations a more complex ownership and associated role system is included. Available roles are "creator", "master", "helper", "user", "listener", and "applicant"[25], each with a varying degree of access to and control over conversations. Some additional options related to security and confidentiality are focused on user privacy such as blocking certain users, whether video capabilities are shown to other users, availability status can be shown on the web, and from whom calls and messages can be received.
- **Scalability:** The current member limit for conversations is 300 participants[26]. However it is hard to imagine a conversation with more than a dozen active participants given that with 300 people simultaneously typing something the chat window would be filled with 300 lines of "is typing..notifications.
- **Version tracking:** As the main artifacts in Skype are messages rather than a single consistent document there is no traditional version control. Rather every message has an associated timestamp and appears in a linear fashion in a conversation. One functionality which mimics a commonly found component of version tracking in other collaboration systems and is also available in Skype is a notification system. Users can either choose to be informed about every incoming message in a conversation, set up notifications for specific terms or turn off notifications altogether.
- **Visualizations:** Skype does not include any features to visualize collaboration activities such as messages and conversations.

[25]Skype: What are chat commands and roles?. `https://support.skype.com/en/faq/FA10042/what-are-chat-commands-and-roles` (Last access: 20.07.2012)
[26]Type */info* in a conversation to see the number of current participants and the current member limit.

6 Overview of Findings

Table 1 provides an overview of the findings obtained in the reviews in chapter 5 whereby an 'x' denotes that a certain design dimension is included in the system. The overview is intended to be a baseline for the analysis and design of collaboration systems.

Among other things it shows that a system which isn't intended for real time use such as MediaWiki also doesn't provide any awareness capabilities as these are apparently not deemed necessary. Unlike the other four systems it also doesn't provide an ownership system which could be an area for potential enhancements in the future.

When it comes to mobility neither EtherPad nor MediaWiki currently provide dedicated support while GitHub's mobile apps are very limited in their functionality. This indicates opportunities for future design and implementation work for these three projects. EtherPad also doesn't use locking mechanisms though it might be worth exploring such an integration both from a practical and research perspective. With Skype the table shows that no form of concurrency control or locking mechanisms is included which can be interpreted as an indication that they aren't needed in such communication-focused products.

Last but not least the overview also shows that Google Docs implements a very broad range of options in all design dimensions. From that perspective it is the most comprehensive collaboration service among the ones reviewed in this thesis.

Overall it is clear that covering every design dimension might neither be necessary nor desirable for every type of collaboration system. However the overview also shows that some systems barely include some of the dimensions and at least considering their inclusion might be a worthy analysis and design exercise.

Design Dimension	Etherpad	GitHub	Google Docs	MediaWiki	Skype
Time					
non real time	x	x	x	x	x
real time	x	x	x		x
Space					
same space	x	x	x	x	x
different space	x	x	x	x	x
Awareness					
availability status					x
current activity / focus			x		x
communication tools	x	x	x		x
Granularity					
individual characters	x		x		
units		x	x	x	x
compositions		x	x		
entire artifacts	x	x	x	x	x
Locking					
pessimistic concurrency control					
optimistic concurrency control	x	x	x	x	
explicit locking		x	x		
implicit locking		x		x	
remove or steal locks					
Merge Transparency & Forking					
transparent merging	x	x	x	x	
expose merging mechanisms		x		x	
Mobility					
adaptive user interfaces			x		x
limited functionality designs		x	x		x
support for offline use			x		
designing for interruptions			x		
seamless switching of devices			x		x
Ownership and Security					
levels of ownership		x	x		x
transfer of ownership	x	x	x		x
confidentiality	x	x	x		x
Scalability	15	unlimited	50	-	300
Version Tracking					
actual differences	x	x	x	x	
attach metadata		x	x	x	
notification system		x	x	x	x
revision control	x	x	x	x	
Visualization	x	x			

Table 1: Overview of the design space and the reviewed collaboration systems.

7 Conclusions and Future Work

This chapter sums up the conclusions of this thesis and provides an overview of areas for possible future work.

7.1 Conclusions

The results presented in the previous chapters align with the scope laid out in chapter 1. A design space for user interface design in collaborative systems was developed. Subsequently it was used to review five publicly available collaboration systems. By analyzing and comparing the different approaches this design space and the reviews provide a foundation for the analysis of current and the design of future collaborative systems.

Contrary to other existing design spaces and similar classifications this thesis put a bigger emphasis on user interface design. By including design options related to the mobile use of collaboration software, it also took the more recent but ever-increasing importance of ICT supported collaboration outside the traditional office context into consideration.

In terms of its application this design space is believed to be particularly relevant when analyzing existing collaboration systems or designing new ones. For an analysis the design categories and their options can be understood as questions such as "How does the system communicate awareness information?". When used during the design they can be understood as "How should the system be designed to communicate awareness information?". Such questions seem to be particularly relevant during the early stages of the software design process (e.g., requirements engineering, conceptual work modeling, and early screen designs).

7.2 Contributions

In chapter 2 the thesis provided an introduction to the history of the development and research of collaboration systems since the 1960s. It highlighted some of the long-established undercurrents within this area of research such as its strong focus on work and work environments. It also looked at some fairly new and ongoing trends such as the broader adoption of collaboration systems in the past few years.

The different methods which were used for creating the design space and reviews were detailed in chapter 3: literature research, analyses and reviews of five existing collaboration products and services, and the iterative compilation of the design space.

The design space which represents the core outcome of the thesis was presented in chapter 4. It consists of the dimensions *time*, *space*, *awareness*, *granularity*, *locking*, *merge transparency and forking*, *mobility*, *ownership and security*, *scalability*, *version tracking*, and *visualizations*. The chapter also contains the known limitations and an outlook towards current developments which might impact the design space in the near future.

The reviews of EtherPad, GitHub, Google Docs, MediaWiki, and Skype were documented in chapter 5. An overview of the results and an explanation of how to interpret and use them in the analysis and design of collaboration systems was provided in chapter 6.

Finally possible topics for future work are presented below.

7.3 Future Work

With regard to future work there are a number of areas which merit further attention. First and foremost it is necessary to evaluate the design space's completeness, acceptability, and practical usefulness in the context of software development projects. One way to achieve this would be by discussing the findings and their implications with designers and engineers involved in the analysis and design of collaboration systems. Moreover it is proposed that a combination of the thesis' findings with such discussions could serve as the basis of a deck of cards similar to the IDEA Method Cards[27]. Such cards would provide a practical tool to support the analysis and design of collaboration systems.

Secondly the results of the reviews provide a foundation for exploring design combinations which may not have been developed yet. Such new systems or enhancements of existing systems could lead to interesting new forms of collaboration with both technical as well as social implications. For example what would an online knowledge resource such as Wikipedia be like if its underlying MediaWiki software supported real time collaboration with dozens of simultaneous editors of articles?

Last but not least each dimension of the design space and its options could be the subject of further in-depth investigations. Many of them have already been studied since the early days of collaboration related research. However the ongoing move towards more mobile and ubiquitous collaboration scenarios, new technologies, and the associated changes in user expectations create a significant potential for new and different perspectives of these dimensions.

[27]IDEO: Method cards for IDEO. `http://www.ideo.com/work/method-cards/` (Last access: 01.04.2013)

8 References

[Ahmed und Tripathi 2010] Ahmed, T. ; Tripathi, A.R.: Security Policies in Distributed CSCW and Workflow Systems. In: *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on* 40 (2010), nov., Nr. 6, S. 1220 –1231. – ISSN 1083-4427

[Crabtree u. a. 2005] Crabtree, Andy ; Rodden, Tom ; Benford, Steve: Moving with the Times: IT Research and the Boundaries of CSCW. In: *Computer Supported Cooperative Work* 14 (2005), Nr. 3, S. 217–251. – URL `http://www.springerlink.com/index/10.1007/s10606-005-3642-x`

[Dourish und Bellotti 1992] Dourish, Paul ; Bellotti, Victoria: Awareness and coordination in shared workspaces. In: *Proceedings of the 1992 ACM conference on Computersupported cooperative work CSCW 92* 3 (1992), Nr. November, S. 107–114. – URL `http://portal.acm.org/citation.cfm?doid=143457.143468`

[Dunlop und Brewster 2002] Dunlop, Mark ; Brewster, Stephen: The Challenge of Mobile Devices for Human Computer Interaction. In: *Personal and Ubiquitous Computing* 6 (2002), S. 235–236. – URL `http://dx.doi.org/10.1007/s007790200022`. – 10.1007/s007790200022. – ISSN 1617-4909

[Greif 1988] Greif, Irene: *Computer-Supported Cooperative Work: A Book of Readings.* Morgan Kaufmann, 1988. – 783 S. – URL `http://www.amazon.com/dp/0934613575`

[Grudin 1994] Grudin, Jonathan: CSCW: history and focus. In: *IEEE Computer* 27 (1994), Nr. 5, S. 19–26. – URL `https://research.microsoft.com/users/jgrudin/past/Papers/IEEE94/IEEEComplastsub.html`

[Hincapié-Ramos u. a. 2011] Hincapié-Ramos, Juan D. ; Voida, Stephen ; Mark, Gloria: A design space analysis of availability-sharing systems. In: *Proceedings of the 2011 ACM Symposium on User Interface Software and Technology* Bd. 1, URL `http://dx.doi.org/10.1145/2047196.2047207`, 2011, S. 85–96

[Johansen 1988] Johansen, Robert: *GroupWare: Computer Support for Business Teams.* New York, NY, USA : The Free Press, 1988. – ISBN 0029164915

[Johnson-Lenz und Johnson-Lenz 1998] Johnson-Lenz, Peter ; Johnson-Lenz, Trudy: Groupware: coining and defining it. In: *SIGGROUP Bull.* 19 (1998), August, S. 34–. – URL `http://doi.acm.org/10.1145/290575.290585`

[MacLean u. a. 1996] MacLean, Allan ; Young, Richard M. ; Bellotti, Victoria M. E. ; Moran, Thomas P.: Questions, options, and criteria: elements of design space analysis. In: Moran, Thomas P. (Hrsg.) ; Carroll, John M. (Hrsg.): *Design rationale.* Hillsdale, NJ, USA : L. Erlbaum Associates Inc., 1996, S. 53–105. – URL `http://dl.acm.org/citation.cfm?id=261685.261707`. – ISBN 0-8058-1567-8

[Menascé und Nakanishi 1982] Menascé, Daniel A. ; Nakanishi, Tatuo: Optimistic versus pessimistic concurrency control mechanisms in database management systems. In: *Information Systems* 7 (1982), Nr. 1, S. 13 – 27. – URL `http://www.sciencedirect.com/science/article/pii/0306437982900035`. – ISSN 0306-4379

[Penichet u. a. 2007] Penichet, V.M.R. ; Marin, I. ; Gallud, J.A. ; Lozano, M.D. ; Tesoriero, R.: A Classification Method for CSCW Systems. In: *Electronic Notes in Theoretical Computer Science* 168 (2007), Nr. 0, S. 237 – 247. – URL `http://www.sciencedirect.com/science/article/pii/S1571066107000394`. – ¡ce:title¿Proceedings of the Second International Workshop on Views on Designing Complex Architectures (VODCA 2006)¡/ce:title¿. – ISSN 1571-0661

[Rama und Bishop 2006] Rama, Jiten ; Bishop, Judith: A survey and comparison of CSCW groupware applications. In: *Proceedings of the 2006 annual research conference of the South African institute of computer scientists and information technologists on IT research in developing couuntries - SAICSIT '06* (2006), S. 198–205. – URL `http://portal.acm.org/citation.cfm?doid=1216262.1216284`. ISBN 1595935673

[Rittenbruch und Mcewan 2009] Rittenbruch, Markus ; Mcewan, Gregor ; Markopoulos, Panos (Hrsg.) ; De Ruyter, Boris (Hrsg.) ; Mackay, Wendy (Hrsg.): *An Historical Reflection of Awareness in Collaboration*. London : Springer London, 2009 (Human-Computer Interaction Series). – 3–48 S. – URL `http://www.springerlink.com/index/10.1007/978-1-84882-477-5`. – ISBN 978-1-84882-476-8

[Schmidt 2002] Schmidt, Kjeld: The Problem with 'Awareness': Introductory Remarks on 'Awareness in CSCW'. In: *Computer Supported Cooperative Work (CSCW)* 11 (2002), S. 285–298. – URL `http://dx.doi.org/10.1023/A:1021272909573`

9 List of Figures

10 List of Tables