Adding social annotations to Electronic documents

Daniel Wenner

Masteroppgave
Master i Teknologi - Medieteknikk
30 ECTS
Avdeling for informatikk og medieteknikk
Høgskolen i Gjøvik, 2008

Avdeling for
informatikk og medieteknikk
Høgskolen i Gjøvik
Postboks 191
2802 Gjøvik

Department of Computer Science
and Media Technology
Gjøvik University College
Box 191
N-2802 Gjøvik
Norway

Abstract

While annotations have been common in normal books for a long time it has never been a big hit in electronic documents. In normal books it has been proven that it can be useful to make annotations and to read other peoples old annotations. As electronic learning material is becoming more and more common a positive effect of adding such shared annotations should also be possible there.

This thesis explores what attributes such a system would need to maximize the utility readers get from an annotated text and the technical challenges with implementing such a system. As previous studies have not found any conclusive results about the effect that sharing annotations have on readers this is also investigated by trying an implemented system on users. The fact that annotations change the structure of a documents based on how users annotate it is also discussed and suggestions are made to potential useful applications of this new structure.

Based on a structured analysis of previous research and other implementations a set of attributes important for a shared annotation system are presented. A prototype system that attempts to implement a set of these attributes was also created to get insight into the technical challenges encountered. A set of common challenges were found and solved, with a solution for asynchronously updating annotations being the most novel solution found, but the whole system is a good example of a mash-up of different solutions.

Users were also interviewed about the use of the system and they are all positive about the implications of a sharing annotations and think it will make it easier to effectively find additional material about a subject. As previous studies this indicates that students have at least a positive view on using such a system.

Preface

This MSc thesis is the conclusive work after studying media technology for two years at Gjøvik University College. The topic was found during the pre-project for the master thesis and was chosen because of a general interest in electronic publishing and social collaboration projects.

I would like to thank Rune Hjelsvold for excellent guidance and encouragement while writing this thesis. Thanks also go out to the students that were willing to use the system and give me attention while I was presenting it. A special thanks go to those among the students that participated in the interviews with their time and insightful answers.

My fellow master students in both the media technology and information security class were also a great help to make the writing of the thesis feel as a less lonely activity.

Contents

List of Figures

List of Tables

Listings

1 Introduction

Annotations are pieces of information attached to a specific place some form of document. It is traditionally used in connection with draft documents, comments to computer code, annotated laws or to add marginalia to text books. It is also a practice used by students when they are reading educational books to add their own notes or highlights to the text they are reading. There are different reasons for doing this. Some do it keep track of where they are in the text, some to know the important parts of the text for later rereading and some even look for used books with usefully marginalia from previous owners [1].

As parts of the curriculum in many subjects are moving away from paper books to online or digital resources there is a need to add similar functions to digital documents. Digital documents such as web pages has a great advantage over traditional paper books in that they are not physical copies of text and can therefore be identically copied with almost zero cost. The internet also allows for sharing information to a degree unprecedented in human history and a willingness for sharing has been shown by shown by various “web 2.0” communities and services such as photo sharing community flickr, the bookmark sharing site del.icio.us and the community encyclopaedia Wikipedia.

The purpose of this thesis is to look at solutions for sharing annotations with others reading the same document over the internet. Because it is not a physical copy it is possible to create additional data that all the readers of the material has access too. The thesis is focused on readers of educational material and the hypothesis is that giving them the ability to annotate the material will increase the original material’s usefulness and allow for collaboration between students. The thesis also focuses on how a system for accomplishing this was implemented and discusses what features are usefully to support reading.

1.1 Problem area

There are several annotation systems for different purposes in use today. The most widely used ones are probably the annotation functions in Microsoft Word and Adobe Acrobat. Those are both systems where annotations are stored with the document and are only available to other users that exchange that same file. In this project however the focus is on systems that enable users to share annotations not just with users that exchange the same the document, but also users that view the same web page. Annotations that are written with the knowledge that other people will read them have been shown to be written with more intent from the author [2]. Sharing annotations for use by other’s was also envisioned at least as early as in Vannevar Bush’s famous essay “As We May Think”[3] from 1945 where he describes the memex, a system to allow cross references between different books and films from a library. The memex concept included an idea for annotation about items contained in the library that could be copied and shared to other people’s memex machine.

How an annotation system should behave is also looked into in this thesis. For web annotations there are two main approaches. One solution is to import documents into the system itself and then convert them to some internal system and then let users share comments within the system. Examples of this can be found in The processed book[4], co-ment and the HyLighter[5] systems. The other major approach and the one chosen in this thesis is to allow the system to be superimposed over any web page and let subsequent visitors see the same annotations. This is used in many systems that let users put post-it style comments or highlights on web pages. Examples of systems like this is Diigo.com[6] and W3C's annotation solution the Annotea Project[7].

As the student is reading educational material the main focus needs to be on the the curriculum itself. An annotation system should be there to support the reading and not get in the way of it. This means that the system need to be relatively easy to invoke, and at the same time powerful enough to easily find or make the wanted annotation. It should also be easy to hide or filter out annotations that get in the way of reading and try to adhere to such guidelines set fourth by the likes EBONI project [8] to keep the original text readable.

As users make annotations on a page the structure of that document changes slightly. It is no longer just the text and other elements the author included, but also includes additional information in the form of annotations. This new structure can tell us something about where the important information in the document is found.

1.2 Motivation and benefits

In the early beginnings of the web there were some services that tried to implement different kinds of annotation services. After the market downturn around 2001 many of these services disappeared or were discontinued because of lack of funding such as Third Voice[1] or just seem discontinued such as CritLink [9]. Group annotations were also implemented in the Mosaic browser in version 1.2, but all that is left of this project are notes about it on scattered web pages[2]. However as browser's have become more mature new systems are starting to emerge that take up the heritage of the dead or older systems. Foremost of these systems is perhaps Diigo.com [6] a service that enables both social bookmarking and annotation facilities. As there seems to be a new push to get annotation systems accepted as a usefully tool for research or to add sticky notes to web pages this is a good time to look at what makes group annotation systems work or why they don't.

This thesis is focused on looking of the effect of making annotations shared among all who use the system. The implication is that everyone that can view the web page can also view the annotations as long as they have the annotation system loaded. This creates a new dynamic in the page where it is not only the author that has a say in what the page says. The users can also contribute with relevant comments, highlight important items or link to further information about something. The hypothesis is that this will increase the usefulness of the document by allowing user themselves to contribute extra information in addition to that originally contained in the page.

The effect of such sharing on users has been studied previously, but no conclusive effects have yet been reached. It is the hope that thesis can get a look at how users react

[1] Wired article about the discontinuation of Third Voice titled:"Third Voice Trails Off...." `http://www.wired.com/techbiz/media/news/2001/04/42803`

[2] Paul Perry's page about Web Annotations that seems to have last update in July 2001 is full of links to dead systems and mentions mosaic 1.2's features: `http://www.paulperry.net/notes/annotations.asp` Last visit: 29/5/2008

to such sharing of information and what features are needed to support it.

1.3 Research questions

The research question is divided into two groups, the main research question and four sub-questions that must be answered to answer the main question.

1.3.1 The main research question

How can we design a system for sharing annotations on a global scale that increases the utility and learning benefit of electronic learning material and how will users use such a system?

1.3.2 The sub questions

1. What attributes are important for a functioning annotation system to support and extend reading and understating of electronic reading material?

2. How does shared annotations affect the reading and involvement with learning material?

3. What technical challenges exist when implementing a system for sharing annotations?

4. How can annotations generate a richer structure for electronic documents and how can this be utilized?

1.4 Method

1.4.1 First sub question

To answer the first research question a survey of related systems and literature has been done. With this as a base a TRIZ problem analysis was done to uncover what the important properties that needed to be in an annotation system needed to be. The implementation of the prototype itself and the implementation chapter 5 is also a part of answering this question to see if it was possible to implement a solution that users could use to test the later research questions.

1.4.2 Second sub question

This question is answered by looking at answers from users after testing the system, the data collected from the annotations and their use of the system. Related literature and conclusions from related material is also used to discuss the implications of including an annotation system.

1.4.3 Third sub question

The technical difficulties are uncovered by the study of related work and annotation systems. The implementation of a prototype is used to demonstrate how the technical challenges can be solved.

1.4.4 Fourth sub question

To answer this a qualitative evaluation of the prototype system and the personal experience gained through study of other annotation research has been utilized.

1.5 Definitions

Annotation Extra information to a text that is situated at a specific place in a document

Orphaning When an annotation can't be positioned correctly in the page it is said to be orphaned. This means that it won't appear in the page, or if the system tries to calculate it's position it could be placed in a spot the user did not intend

Marginalia Text written or placed in the margin of some form of document. Usually as a comment to text nearby

Bookmarklet Technology to put JavaScript code in a bookmark. This enables the piece of JavaScript code to run on any web page

Web 2.0 A term used to coin the emergence of new type of web services that focus on communities

Triz A model based approach to generate inventive ideas and solutions for problem solving.

2 Related work

2.1 Annotation in normal books

An often referenced study on how people use annotations is C. Marshall's [1] study of used books from a university book store. She shows that annotations found in old books are useful to new owners because they add context and often cross references in the text. She also sets up some functions that digital annotation systems should include based on the finding from the study of annotations in paper books. Some of the suggestions are:

- The markings should be distinguishable from the source material
- Should be able to let users make many different kinds of annotations to support individuality
- Smooth transition between public and private annotation
- Integration with reading as an activity. Annotation should interrupt reading as little as possible

In a further study [10] she makes predictions on how hypertext systems will make use of informal annotations based on further study of used text books and additionally by observing readers and doing informal interviews when they were looking through annotated text books. In this study it was found that annotators can create a more hypertext like structure to the book by looking at where users are in consensus about important parts of the book. This can also be used to summarize the content.

In a study to figure out how annotations affect readers Wolf found that readers will read notes differently based on who are writing the notes and that recall is higher on information that is highlighted [11]. He also makes some suggestions for how digitalized books should present annotations: they should be searchable by position, be able to be filtered by type of annotation and the background of the annotator should be presented.

2.2 Research on and implementations of shared annotation systems

W3C has some work on an annotation server called Annotea that can store and share annotations and bookmarks based on RDF and XPointer [7]. This might be the closest we have to a standardized way to make annotations on HTML documents, but the project does not seem to have developed much since 2003. Some clients such as Annozilla seem to be in some form of development still. Everyone can see all other comments about that page on the current Annotea server using Annozilla.

The WebAnn system used by C. Marshall and Brush [2] in a from 2004 was a plug-in for internet Explorer that allows annotations to be made from a sidebar. It was used for a study to find if people used the same annotations when they were published as when they were making private notes. Their findings conclude that only a small fraction of the notes were made public when users had the option to choose and that:

> The content of the personal annotations is apt to be clarified and extended and the anchors are apt to be tidied up to more accurately reflect the scope of the comment; a shared annotation reflects far more authorial intent.

EDUCOSM [12] was as system developed to do annotations of web document using a website as a proxy, allowing users to right click text to add comments to that part of the page. Students worked in groups of two that shared annotations. They found that all students in the test agreed that the ability to make annotations added to the value to the learning process, but that they had the most use of their own annotations. The students who made most annotations also got the best grades. That is the same result as in [13] where the students also read more after the inclusion of an annotation and discussion system. This study also did some predictions on the students grades based on what the students annotated and could predict grades with 75.5% accuracy.

Another similar study was done by Chisik and Kaplan with girls aged 10-12 in the Corvallis study with a system called Alph[14] The girl annotated a story by using emoticons and comment text. The researchers found that:

> Their notes reveal an attachment to the process of reading the novel but an equal interest in sharing their reactions and responses with other readers.

In another study by the same authors [15] with teens aged 10-14 and their parents where comments could made and shared between members of a reading group. This study had more focus on how an annotation affects the reading experience. They found that if there is a perceived social benefit, comments will be made public. Comments in their project were more for fun than for learning, so the perceived social "penalty" for writing a "wrong" comment was low.

A system called HyLighter has also been tried on users by G. Lebow and W. Lick [5] but they found no conclusive results. They make a suggestion that:

> From another perspective, HyLighter provides an effective alternative to threaded-discussion groups. Discussions supported by HyLighter are anchored in the context of a shared text and tend to be more convergent than in threaded-discussions, and, as a result, more productive.

The system works as a web application where books/documents can be imported and shared between invited users. This gives access control to who can see and annotate the document.

A small survey and a system for sharing annotations is made by Glover et al. in [16] where they discuss some of the features from different systems. Based on the survey of other systems they present some *essential* and some *desirable* factors that they want in their system. The factors are again divided in *conceptual* and *technical* factors. The comparison matrix with other systems is presented in figure 1 and a list of the factors is presented below. A more detailed list with descriptions can be found in 7.1.2 where other features that can also be useful are examined.

The essential conceptual factors

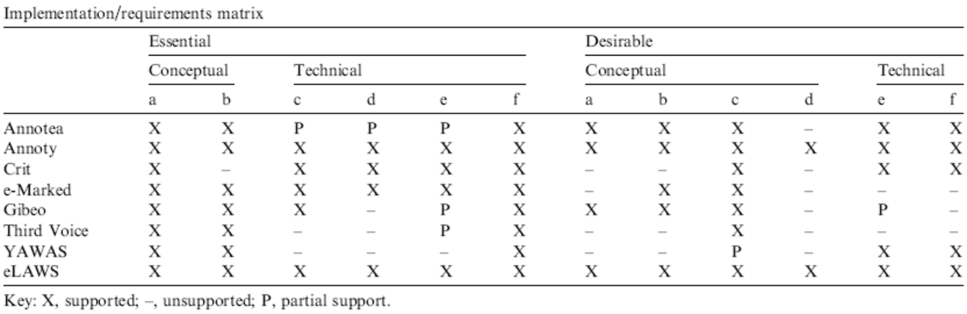
a) Text annotations

b) Private annotations

The essential technical factors

c) No additional software necessary

d) Accessible



Implementation/requirements matrix

	Essential						Desirable					
	Conceptual		Technical				Conceptual				Technical	
	a	b	c	d	e	f	a	b	c	d	e	f
Annotea	X	X	P	P	P	X	X	X	X	–	X	X
Annoty	X	X	X	X	X	X	X	X	X	X	X	X
Crit	X	–	X	X	X	X	–	–	X	–	X	X
e-Marked	X	X	X	X	X	X	–	X	X	–	–	–
Gibeo	X	X	X	–	P	X	X	X	X	–	P	–
Third Voice	X	X	–	–	P	X	–	–	X	–	–	–
YAWAS	X	X	–	–	–	X	–	–	P	–	X	X
eLAWS	X	X	X	X	X	X	X	X	X	X	X	X

Key: X, supported; –, unsupported; P, partial support.

Figure 1: The feature requirement matrix for different annotation systems

e) Browser independent

f) Maintains page integrity

The desirable conceptual factors

a) Graphical annotations

b) Linking annotations

c) Shared annotations

d) Annotation reports

The desirable technical factors

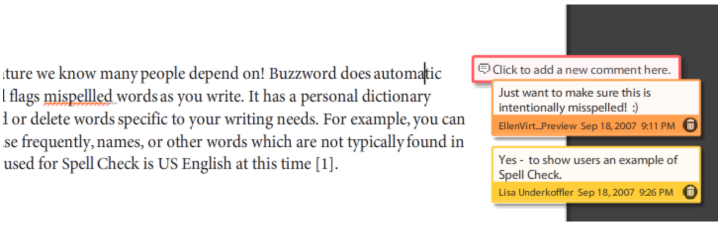
e) Open architecture

f) Non-commercial

The system they developed based on this was called eLaws and could be used with just a web browser, it was an improvement over an earlier system by the authors called Annoty that required Java on the client. From the articles description eLaws seems to be a server side proxy system where other web pages are loaded into the eLaws system and can then be annotated using a sidebar. Reports of annotations can be used to filter annotations based on users or pages they are made on. They found that almost half of the students they tested Annoty on would use it for personal research, 87% for school work and 91% "agree" or "strongley agree" that it had improved their learning on the subject.

Another user test with two different classes taking the same subject was carried out by Wang et al in [17]. They found that annotations helped students study in a more consistent manner and the students taking annotations beat the others on early tests, but on the final exam the results were nearly tied. They attribute this to high motivation to do well on the final exam and

> A ceiling effect due to high motivation to study for final exams no matter what kind of tools or scenarios were provided.



ture we know many people depend on! Buzzword does automatic
l flags mispellled words as you write. It has a personal dictionary
l or delete words specific to your writing needs. For example, you can
se frequently, names, or other words which are not typically found in
used for Spell Check is US English at this time [1].

Click to add a new comment here.

Just want to make sure this is intentionally misspelled! :)
EllenVirt...Preview Sep 18, 2007 9:11 PM

Yes - to show users an example of Spell Check.
Lisa Underkoffler Sep 18, 2007 9:26 PM

Figure 2: Example of comments in Buzzword. Comment boxes are added at same line marker is in

2.3 Reasearch in social navigation.

Annotations and how a user moves through a corpus can be used for providing navigation mechanisms to other users. This is based on the fact that users often like to follow the footprints of others. This was used by Brusilovsky et al. in [18], where a system for organising topics was improved by adding colour coding to topics most often visited. What students found lacking in this system was a way to also rate the links, and not just base them on number of people following a trail.

Cabanac et al. created a system called TafAnnote [19] that adds a social validation component to annotations. Threaded discussions are enables on the annotations and users can post replies to the annotation either agreeing or disagreeing to the annotation posted. Annotations that are have users agreeing with the annotation boost the social validation and makes the icon representing the annotation more pronounced. They have not done user testing on the system, but plans to test it on users in the future.

2.4 Implementations of annotations for documents or e-books

Many systems exist for writing documents that can also make annotations about the document, for example Microsoft Word or Adobe Acrobat, but they are limited to sharing with only a few people who are working on or reading the same copy of the document and are embedded in the document itself.

Some E-book readers and programs can also provide annotation features such as the VitalSource Bookshelf software [20] which is a commercial digital library and e-book software. It's however limited to sharing annotation only with those you specify with an e-mail address.

With the advent of web based word processors like Google docs and Adobe's Buzzword sharing documents and collaboration on them is made a bit easier as several users can edit, view and comment on documents at the same time. Buzzword can share your documents with users that you provide the email address to. An example of commenting in Buzzword can be seen in figure 2.4.

Processed book: Open source server called PBOS that show annotations in normal web browser. Books are imported to internal wiki format in a database and annotation stored in another table. Includes mechanisms to browse and find documents statistics. The system was created as a prototype of a system described in what is in the makers of the software's own words a speculative essay [4] about the future of electronic texts and how to enhance reader involvement with the text. Annotations can be of different types like link, comment, incoming link and new types of annotations can also be constructed by users. Annotations are placed as a short note in the margin about the type of annotation. The user can decide the look of this note themselves by changing colours.

2.5 Web based social services

Other types of services that let users contribute by categorizing content by adding tags or otherwise adding their own content are plentiful on the net. By aggregating the user metadata what has been called folksonomies are formed. del.icio.us and flickr are perhaps the most famus examples of this. Research has recently gone into figuring out why people contribute [21], why such systems work and how statistics on the results can be used to semantically annotate the web resources [22].

2.6 Web annotation systems

Microsoft previously had a system for web annotations called “Web Discussions” that shipped with office 2000 and 2003. A large user study of work on collaborative specifications for software components within Microsoft was done by Cadiz et al. in [23]. In their tests “the primary reason that people stopped using the system was annotation orphaning”. They also find that making annotations is not a good way to make people react quickly to issues as the participants did not check updates to the annotation discussions regularly. It was however a nice way to discuss minor issues and freed up time for important issues at meetings. They blame this on a notification system for annotations that did not give good enough feedback on who made the annotations, where in the document they were made or if there were new annotations. “In addition, people said they did not make annotations when they felt a comment could be taken the wrong way, or when they did not want to appear overly harsh.” They also conclude that annotations are often not rich enough to support discussions in a collaborative environment as only a small part of annotations in their test had follow ups.

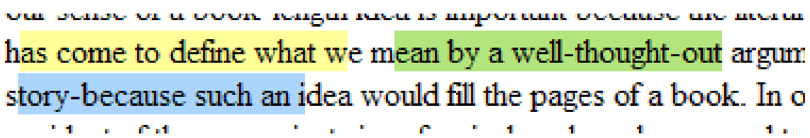
2.6.1 List of some other web annotation systems with description

As mentioned in the introduction the perhaps most comprehensive system that let users share annotation among all users of the system is diigo.com. It calls itself a Social Information Network and focuses on providing everything users need to organize and remember important parts of pages. It lets user annotate any web page and share the notes to everyone or to a specified group. It can also make slide shows of several annotated pages. Also provided is a bookmark tool in the same way as del.icio.us(and it can also import and export to del.icios.us) and allows you to give tags to page bookmarks. It can be accessed through a Firefox, Flock or Internet Explorer add-on toolbar or through a bookmarklet in other browsers.

Co-ment.net imports text into an internal system and lets a group of users collaborate on the document. Includes a search filter to filter out annotations by user or text contained in the annotation. Features to decide the role of users in relation to the document is also there, so some can edit the document and others just view or annotate it. Also includes a feature for replies to comments to create a discussion within one annotation.

Gibeo.net is a commercial service that was loaded by using the gibeo.net site as a proxy. It could be loaded by inserting `.gibeo.net` after the the host in a URL to enable annotation features. It does not seem to work anymore for annotations even though some parts of the site still work, and has seemingly not been updated since 2004.

Zezame.com is a commercial system that works by using a bookmarklet on the client side. When the user pressed the bookmarklet the current web-page is downloaded to the Zezame server and the user is redirected to the Zezame modified version that enables



has come to define what we mean by a well-thought-out argum
story-because such an idea would fill the pages of a book. In o

Figure 3: Diigo higlights. Different colours for shared, private and other peoples annotations.

notes to be made on the page. Has administration over what users can add comments to a document. The Zezame download seems to be unavailable at the moment, but the web page is still there, so this is based on the descriptions given on the page.

Shiftspace.org is an open source project to enable marking on web pages. It works by loading the system into any web-page using a Greasemonkey[1] script and then stores the markings called "shifts" on a Shiftspace server. All shifts on a page can be turned on and of with a menu at the bottom of the screen. This is dependant on Greasemonkey being able to use it's own brand of AJAX communication, so it won't work with current versions of Greasemonkey where this is disabled for security reasons.

Wikalong.org adds a Wiki in a sidebar that can be used to make comments, notes or have a discussion about the current page. Implemented as a plugin for Firefox.

Fleck.com uses the Fleck.com homepage as a proxy to browse web-pages with the Fleck annotation system loaded. It loads as a flash overlay on the entire page to enable it's note system. Allows adding of shared post-it notes on the pages loaded through Fleck. Trailfire.com can make a path of annotations through several pages and is usefull for organizing pages and their comments by a topic. Comments appear as a post-it note on the page that can be minimized to take less space. Uses the Trailfire.com homepage as a proxy for browsing the annotations when using a bookmarklet, but loads them directly on the page using a Firefox extension.

Stickis.com allows the sharing of post it notes through an Internet Explorer or Firefox plugin.

Sharedcopy.com allows highlighting on any web-page using a bookmarklet. This loads the Sharedcopy system into the page and lets you use the annotation tools available. You can mark text, add simple graphics(line, circle and box) and add notes. After the page is annotated they can be saved to Sharedcopy's servers. The annotated page is saved to Sharedcopy's server as a copy of the original page plus the annotations and can be shown to other's by sharing the URL to that page on Sharedcopy's server. General comments and tags can also be added to pages, so it also functions as a social bookmarking service.

Mystickies.com Enables the user to add post-it notes to web-pages. These can be tagged, but can't be shared with other's in the current version.

2.7 General learning material design guidelines

Some pointers on how to evaluate educational usefulness e-books can be found in [24]. An evaluation framework is set up that can be used to measure educational usefulness and the usability of e-book user interfaces. A set of guidelines for good educational material design has also come out of the EBONI project where surveys and user studies were carried out to find best practices [8].

[1] Greasemonkey is an extension for Firefox that allows custom javaScripts to be run on any web page

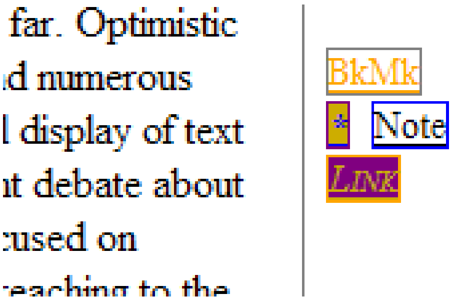
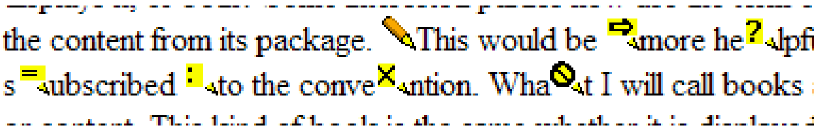


Figure 4: Different types of annotation for Processed book



the content from its package. This would be more he lpfi
s ubscribed to the conve ntion. Wha t I will call books

Figure 5: Annozilla style of annotations

2.8 Visualization and UI design for annotations

Different forms of visualizing annotations

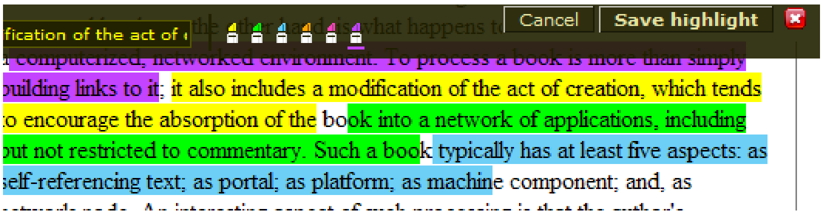
[1] argues for a free-form capability for annotations because annotations are individual in form. The downside of giving users control over annotation appearance is that it can quickly create clutter on the page. If every user annotates using different colours schemes or use different symbols the intention is not easy to grasp for others. Some solutions to this problem are:

- Annozilla solves this visualising of each category of annotation with a different icon. See figure 5

- (Diigo solves it by preselected colour for highlights. Different for shared and private highlights. These can also be customized into a few different styles of display linke underlining instead of highlighting. See figure 3.

- Processed book lets users choose their own colours and background for margin notes. Text of the note is decided by type of annotation. See figure 4

- Shiftspace solves highlights by letting user have choice of colour for highlighter. See figure 6.



computerized, networked environment. To process a book is more than simply
building links to it; it also includes a modification of the act of creation, which tends
to encourage the absorption of the book into a network of applications, including
but not restricted to commentary. Such a book typically has at least five aspects: as
self-referencing text; as portal; as platform; as machine component; and, as

Figure 6: Shiftspace highlight dialogue

2.9 Summary

As seen there are is a lot of previous work done that tries to figure out the effect and best use of annotations. The user test point towards the fact that users want to use an annotation system, but have not come to hard conclusions about how it affects cooperation and learning among the students. As a diverse set of features exist in the many systems reviewed Glover et al. [16] does a good job of summarizing important features, but are some important features missing, like ways to weight the value of an annotation as done in TafAnnote [19] or add discussions as some of the other solutions do.

To investigate the problem in a structured way, the next chapter presents an analysis to find out what the problems and available solutions in the area are.

3 Problem analysis

I this chapter a light TRIZ analysis of the related work is done to figure out what makes a good annotation system and what factors are detrimental to such a system. It follows the suggested method for Triz in service design set forth by Zhang et al. in [25]. The basic method is to create a model of the service by looking at what usefully functions it has and what effects are harmful to achieve an ideal service. An ideal service is one where there are only useful functions and all harmful and detrimental effects are eliminated or minimized.

3.1 Preliminary problem analysis

The first stage of the TRIZ service design process is to identify existing problems and find information about the status of the problem. This is done by answering a series of questions that should help understanding the depth of the original problem [25].

3.1.1 What is the purpose of the target service?

The purpose of the created service is to enable shared commenting on web based learning material. This allows for a collaborative reading experience where readers can share ideas about the material with each other. [16]. This should also increase the general utility of the material and increase the students ability to remember important concepts.

3.1.2 What is the existing problems and the known solutions?

Standardization

One problem with annotations systems is that there is very little standardization in the area. The closest thing to a standard is the Annotea project that is headed by W3C[16][7]. There does exist a few implementations of both servers and clients based on this standard and new ones can be created. However new work on this project looks unlikely as it has not been updated for some time. It is also lacking in features beyond the ability to make annotations.

Open source project like Shiftspace[26], the processed book [4] or eLaws [16] could also contribute to a standard emerging by the nature of the openness of their format of annotations.

Compatibility between different systems could also happen if some of the commercial systems open up their annotations to be used in other services through an API. Among the numerous social bookmark services this is more commonplace and some services like Diigo (that is both a social bookmark and annotation service) and `www.onlywire.com` use those APIs allow you to bookmark to several pages at once. Diigo is also in the process of creating an API to their service but, this only covers the bookmarking functionality yet, but it seems they are planing to expand it.

Hard to place annotations in dynamic content

Placing annotations in a specified spot in a HTML document that might change over time is a difficult problem. The chance of annotation orphaning is high when a document changes. Some solutions like Brush et al's in [27] have been tried to achieve a robust method for placing annotations. They say that positioning is usually done in three different ways:

1. Calculating an offset from characters or x/y offset on the page.
2. Search for the highlighted string.
3. Positioned by calculating a signature for the place in the page that the highlight belongs.

These have different pros and cons when a page changes. The offset based solutions fail quickly if the page changes because some movement of characters will happen. The signature method can work when content is changed as long as the identifier for the content is preserved. A search for the string will also succeed as long as the string does not change, even if it is moved to a different place in the document, but could have several matches in a document.

How do users behave when faced with shared annotations?

When annotations are prepared for sharing C. Marshall found [2] that annotations that they know will be shard show more authorial intent. In a cooperative environment the Microsoft study [23] also showed that people did not comment overly harsh about other peoples comments. This is also mentioned in section 2.6. Wang also found that high achieving and low achieving students annotate the same amount in shared environment [17].

How should annotations be visualised?

Some real world solutions can be found in 2.8. Marshall [1] suggest as much free-form annotations as possible to support individual annotation practices. Annotations should not distract the reader from the text.

How should views of annotations be filtered?

Filtering annotations can be important when you want to find annotations that fit certain criteria. Filtering can also be done by individual annotations as it is done in Shiftspace where annotations are toggled from a menu. While this shows all annotations at the same time, it is a slow way to filter with many annotations on the same page. Some more general criteria to filter by are:

- Tags on the annotation
- Type of annotation
- Date the annotation was made
- By submitter or reviewer
- State of annotation. I.e. only show comments that are unread or not replied too.
- Public or private
- Votes for the annotation or user
- Searching for annotation content

Reach critical mass of users

For shared systems a critical mass of users is need for at least one document for annotations to be useful. A user base is naturally hard to build up and maintain [26]. Many systems seem to have withered because of lack of users.

What kinds of annotations should user be able to make?

There are many different forms of annotations that a system could theoretically support. Some of the things implemented in other systems are:

- Text comments
- Highlights
- Drawing
- Predefined figures
- Audio comments. A problem with this is the “the inability to skim audio”[23])
- Video comments and discussions such as done by Youtube.com
- “post-it” notes
- Discussions in annotations
- Comments to whole page
- Changing of page elements like in Shiftspace.org
- Trails or presentations of one users annotations over several pages like (Diigo or Trailfire)
- Outbound links to other resources
- Links to the annotation itself like in The processed book [4]

Being able to many different types of annotations should help to express the users creativity and let them ind a more individual form as mentioned by C. Marshall[1]. It does however add to the time it takes to develop and maintain the solution and can make a the annotated material drown in different kinds of annotations.

What are the learning benefits of annotations?

In the EDUCOSM all 50 student agree or strongly agree that it helps with learning, but they are unsure of any actual effect more than a just psychological[12] one. More about it can be found in the related literature in 2.2.

Students that annotated showed better performance in initial test and read more initially, but performed the same as a control group on exam in the Hwang et al study [17] More can be found in 2.2

Average time spent reading increase dramatically for annotating students the in Wang’s et al. study[13] and most students are willing to use the system further. They also found that they could predict grades based on the annotation activity of students. More in 2.2

3.1.3 What is the system structure of the target service operations?

There are three basic structures to the annotation systems. One way is to keep annotations separate from the content being annotated and only store the annotations and where they are made like in Annotea and Diigo. The annotation system is somehow loaded into the current web page to enable annotating on that page, usually through a browser plugin or through loading a script into the page.

Some other systems like Hylighter[5] and the Processed Book [4] import the content to annotate into the service itself and lets the user interact with it within that system. Annotations are typically stored separate from the document, but both are stored within the system.

Document based annotation systems like Word and Acrobat where annotations are connected with one copy of the document. To share the annotation one must exchange the same copy of the document.

3.1.4 What is the ideal final result to the original problem?

The ideal system should:

- Take no time to develop or maintain
- Annotations always keep correct position
- Filtering and searching always returns the exact results the user wants
- Annotations don't disrupt normal reading at all
- Annotate with exactly the type of annotation the user wants at all times
- Annotations always appear as the user wants them
- Sharing is only done with exactly the users the annotations author wants to reach

3.1.5 What are the local constraints or limitations?

Limitations in the case of this project is development time, the number of people that can participate and system resources to run to implementation.

3.1.6 What is the objective for this problem solving project?

To produce a system that includes as few cons as possible while retaining positive effects.

3.2 Problem modelling and formulation

A function diagram of a possible web annotation service is presented in figure 7. This is created on the basis of information uncovered in the preliminary problem analysis (3.1). It's built around the primary usefully function witch is to increase the utility of the e-book through annotations. It is then divided into functions that are useful and harmful to this goal and how they influence each other.

Derived from this model a series of problem statements are then created to decompose the problem into several lesser problems. Only a few of the possible problems from the model are included here

1. Find a way to [increase document utility] without increasing [User frustration]
2. Find a way to decrease [Visual clutter] that does not require less [Amount of annotations] and [Different kinds of annotations]

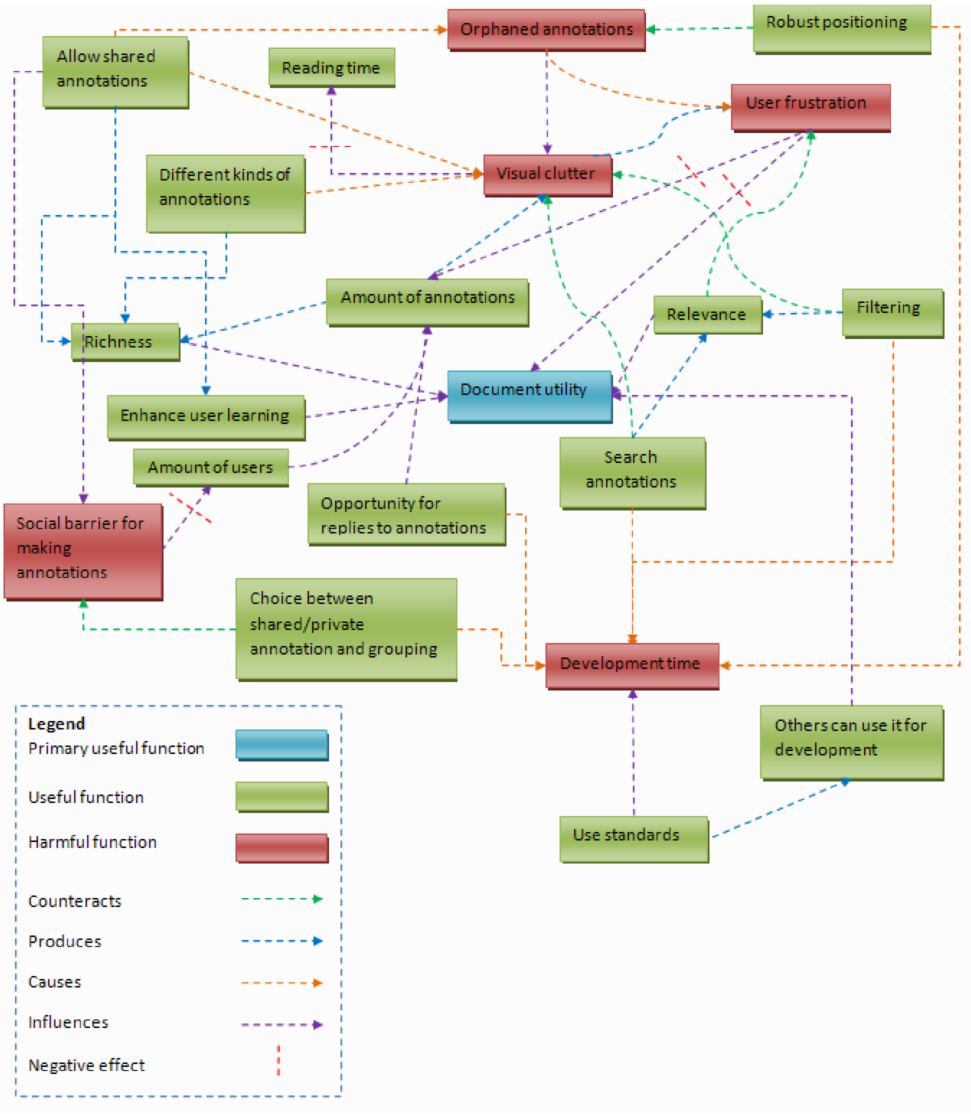


Figure 7: A model of factors influencing usefulness of annotation system

3. Find a way to counteract [Visual clutter] with [Filtering] and [Search annotations] and eliminate the contradiction that it causes [User frustration]. At the same time [Relevance] should increase, but should not use too much [Development time]

4. Find a [Robust positioning] scheme that eliminates [Orphaned annotations], but does not require too much [Development time]

5. Find a way to eliminate the [Social barrier for making annotations] by find an effective way to do [Choice between shared/private annotation and grouping]

6. Find a way to counteract [User frustration] by providing [Relevance] from the [Filtering] and [Search annotations]

7. Find a way to [document utility] that does not require [Relevance], [Richness] or [Enhance user learning] and does not cause [User frustration]

8. Find a way to benefit from [Orphaned annotations], [User frustration], [Visual clutter] and [Social barrier for making annotations]

9. Find a way to eliminate, reduce or prevent the effect of [Visual clutter] on [Reading time]

From the model and some of the possible problems that can be formulated from the model, the main challenges to be solved are:

- Create good filtering and searching possibilities that find the annotations it wants you to find ideas are to allow votes, tags and similar “web 2.0” features to be used for filtering and searching

- Allow users to chose if they want shared or unshared annotations to lower entry barrier

- Have not too complicated, but robust way of placing and keeping annotation position within documents [27]

- Allow different modes of annotations, but not too many to avoid clutter in the document

- As little visual impact on the book as possible to not change normal reading experience, but still have easy access to annotations

3.3 Contradiction analysis and Contradiction elimination

A contradiction in the problem modelling is that to allow all form of annotations and features of an annotation system will give it the most utility, but at the same time make it hard to use and harder to read the text. The same is true for the number of annotations: The more annotations the more utility the system has, but it also has an impact on readability of the text.

Based on some of the applicable 40 inventive principles [28] in this context some solutions can be suggested:

1. *Colour Changes*: As suggested in [5] colours for annotations can change based on the amount of annotations in a text fragment. Different colours are also used there to separate between highlight made by selected users as a form of filtering.

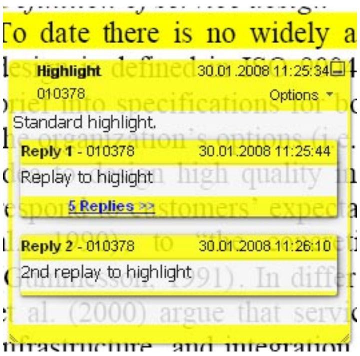
2. *Homogeneity*: Cluster annotations by common properties like date, number of votes or views, annotator, tags or other common properties.

3. *Boosted Interactions*: Get users exited for the system by giving incentive to annotate meaningfully. Reward system or just satisfaction of helping others (e.g. Wikipedia)

4. *Global Quality*: Global sharing. Everyone benefits from annotation on one example.

5. *Nesting*: Allow replies to annotation comments with nested threads. See figure 8.

6. *Preliminary Action*: Look at user needs through what others have found through literature review.



To date there is no widely a

Highlight 30.01.2008 11:25:34
010378 Options
Standard highlight.

Reply 1 - 010378 30.01.2008 11:25:44
Replay to higlight
5 Replies >>

Reply 2 - 010378 30.01.2008 11:26:10
2nd replay to highlight

Figure 8: Adobe Acrobat Professional comments nesting.

7. *Cursing in disguise*: Lots of annotations should be a good thing, but avoid too much annotations on screen at once, and the visual space they consume. Needs a way to filter or unobtrusive way of displaying the annotations.

8. *Feedback*: Use feedback from testers to improve the system before giving it to the users

9. *Direct Contact*: Interviews with participants about the usefulness and experience of sharing the annotations.

10. *Copying*: Use features that have been proven to work earlier other systems.

11. *Separation*: Filter annotations based on time made, tags, groups, annotator or votes

3.4 Solution evaluation and summary

The solutions found and some of insights gained through the problem formulation have been used as a basis for formulating the research questions in 1.3. The goal of the problem analysis and the contradiction elimination is to come as close to the ideal solution as possible. To do this some real world modifications must be made. Filtering and searching will for instance always be an approximation of perfect retrieval. Solutions that have been tried and shown to work with minimal negative effect in other studies should be used.

The problem analysis was used to get a structured look at some of the challenges that faces a potential annotations system. It was used in this project as way to get good ideas as to what was needed of a system for potential use in the test with users. After the problem analysis was done it was clearer what the candidates for such a system lacked and what could potentially be implemented in a prototype to solve some of these problems.

4 System design

Based on the problem analysis and the conclusions reached there a system that manages to include some of the wanted features found was designed. The design of the system tries to include the functions believed to be most useful without compromising it's usability or clutter the original text.

This chapter presents why a new system was created instead of using an established system, details about the architecture of the system and the features wanted in the system.

4.1 Choice of system

The choice to implement an own system instead of using a finished or building further on one of the open source systems was taken to get control over the results from users and to get free reins as to what could be implemented. Implementing the whole system over using something finished does however use up much time for development that could be used for other tasks.

Solutions that were under consideration to be used instead of developing a new system were Diigo, Annotea, ShiftSpace and The processed book.

Diigo does contain most of the wanted features from the system, but is not an open system, so getting the raw results from participants about annotation behaviour would have been a challenge. Diigo also lacked some features when the decision was made, but did get a big upgrade in march that would have made it a stronger candidate for use.

Annotea's strongest point is the fact that it's a W3C project and a such a try to create a standard way to do annotations. It is however not developed any more and lacks support for some features. A client for Annotea would in all likelihood also need to be a browser plugin a stand alone program.

ShiftSpace has a seemingly good open source framework to build on and lets you run your own annotation server. It is however locked to using the extension Greasemonkey and Firefox as a basis and there was initially a desire to create a cross browser solution without the need for plugins.

The processed book is also an open source project, but requires importing documents and staying within the system. Is also limited in what it can do and browser support within the system.

4.2 System architecture

As a system that could be loaded on any page and is not bound to a specific browser by the use of plugins was wanted. A JavaScript solution similar to diigo's bookmarklet and shiftspace's to insert JavaScript into the page the user is viewing was selected for this. More about the solution can be found in the implementation chapter 5.3.

The system is based on a event driven client-server architecture where the the only communication is between the server running the system and the client. A diagram of the system parts can be seen in figure 9. The client in this case is the users web browser that runs the script on a visited page. The server is responsible for communicating with

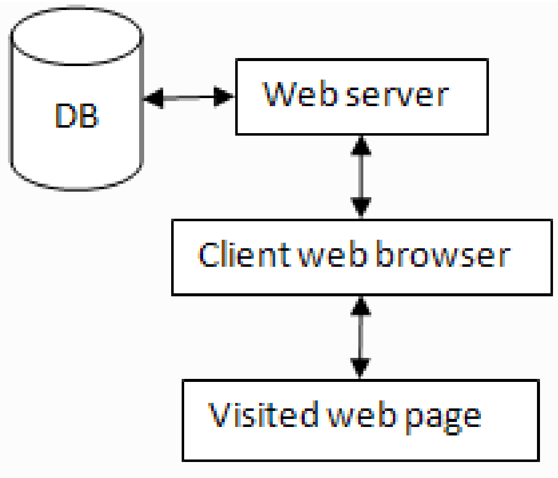


Figure 9: Diagram of the system architechture

the client and storing information in the database. The communication between client and server happens asynchronously when an event takes place on the client(i.e. making an annotation on the web page). This architecture allows all information about an annotation to be stored in the database without the server making contact with the web server hosting the content being annotated.

This way of designing the system is most viable in this project because the web pages with learning material are already created and overlying the system over this allows annotations to made there without moving to a different web page or program. Storing annotations away from the content being annotated and not storing the content itself also sidesteps potential copyright issues.

4.3 Feature list

This section lists some features that would be desirable in the finished system. The feature list is picked by looking at different systems, the state of the art, the problem analys and ideas from web sites that employ ways of filtering comments.

Login system

A login system is need the provide an identity for the annotator. Knowing who made the annotation is important as some people will make higher quality annotations than others. This also allows filtering to annotations made by one person or members of a group.

Connecting usernames with annotations also provides accountability to the writer of the annotations which should lead to more thought out annotations, especially for shared annotations where annotations in general reflect specific communicative intent [2]. User identities also allow positive votes for annotations to be added to the user and be used as an indicator if the user gives useful annotations or not.

Voting

A system for rating annotations was also planned. This could allow for filtering, searching and generation of reports based on rating. Displaying annotations that get more votes differently can also be done like in [19]. In a document with many annotations this can give an immediate visual aid to what's the most important or reliable annotations in the document.

A negative effect of allowing votes is that it could create a barrier for those users who don't want to be "judged" by their peers. On the other hand it gives incentive to write something meaningful to score higher reviews and get satisfaction from this.

Linking and tagging

Supplying links to other resources about something that is not present in the text gives readers the opportunity to check out other sources or explanations.

Tags on annotations can also be used for cross references and a reader created index of the text. It can be assumed that different annotations tagged with same tag are in general describing the same thing and it can be useful to look at that other annotation or area of the text for more info on the subject.

Searching and filtering

Searching for annotations and text in highlighted areas should be possible. This could give more relevant results than a normal full text search as only what users consider useful is marked.

Some things that was planed to be able to filter annotations by was user, groups, time, rating or tags.

Annotation discussions

It should be possible to have discussions within one annotation. This can help correct other's where information is wrong or provide discussion where there is a divided view on something. The number of replies in a discussion can also be used as measure of an interesting annotations either in a positive or negative measure.

Sharing to groups

Sharing to a specified group lets a set of users collaborate. This can be useful for cooperating only those working towards one specific goal or for assigning users roles.

Preserve context

The context of an annotation is important, so it was planed to store not only the text contained in the annotation, but some of the surrounding context. This should have eased retrieval of the correct search result as some of the context can be presented in the results.

Visualizing

As seen in 2.8 there exists some options for visualising annotations. One should quickly be able to tell what sort of highlight an annotation is by looking at the colour, e.g if it's mine or someone else's. Comments and replies should be presented in an information box close to the annotation when the user wants to view them. A menu to use the functions of the system and to provide status information to the users should be used.

5 Implementation of prototype

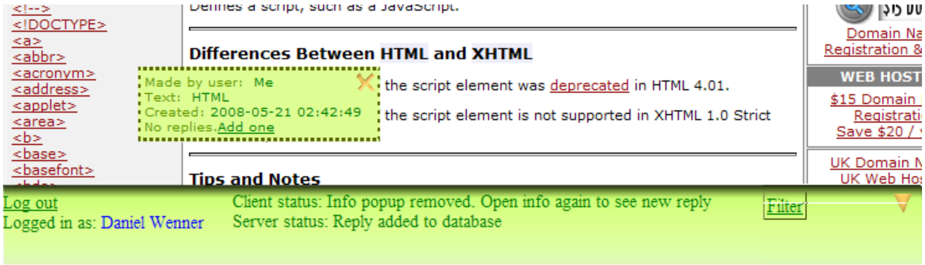
The prototype was made to be able to test out some of the features that the Triz analysis and review of earlier work indicated were things that could be helpfully to have in an annotation system and to try to implement things from the feature list in the system design chapter. It was also the prototype that was tested on users to find out the reactions to certain features and to study annotation behaviour.

This chapter contains details on the implementation of the prototype. It contains some algorithms and descriptions of the important technological building blocks of the system. The reason for choosing certain solutions over others is also included where appropriate.

The first part covers a brief overview of the system and how the user meets it. The descriptions of the system implementation start of with the three problems that were solved before the system could system run at all. These three problems were: How to load the system on any page, how it communicates with the server without the `XMLHttpRequest` and how annotations are positioned within the page.

5.1 Overview

The prototype system has tentatively been called “Dannotation” as a play on the authors name and the word “annotation”. It is a system that runs purely in the users web-browser and has aspirations to be able to run on all pages built with valid HTML. It is loaded into the users browser by clicking on a bookmark that loads the Dannotation system into the current page. From that point the user can view other’s annotations for that page or make their own by selecting a piece of text and choosing an option from the menu that pops up. This allows the user to add just a highlight to the marked text or add a comment, tags, link to external sources or to another annotation on the same page. There is also the opportunity to add replies to any highlight. A basic look of how the system looks can be seen in figure 10. It can also be tried out by going to:
`http://www.stud.hig.no/~010378/thesis/dannotation`.



Defines a script, such as a JavaScript.

Differences Between HTML and XHTML

Made by user: Me
Text: HTML
Created: 2008-05-21 02:42:49
No replies.Add one

the script element was deprecated in HTML 4.01.

the script element is not supported in XHTML 1.0 Strict

Tips and Notes

Logged in as: Daniel Wenner

Client status: Info popup removed. Open info again to see new reply
Server status: Reply added to database

Figure 10: The Dannotation system loaded at w3schools.com

5.1.1 System platform

Dannotation was built using PHP version 4.3.9 for the server side code because this is the version of PHP that runs on HiG’s web-server. Hosting the prototype there was the most viable solution because of plenty bandwidth and good response times. Most of the development was done towards a PHP 5.2.1 server, so it compatible with newer versions as well. As a database MySQL version 4.1.20 is used.

For the client side scripting JavaScript is used. The JavaScript is also augmented by the JavaScript library JQuery because it provides a lot of convenient functions and generally makes the script shorter and more readable.

5.2 Bookmarklet

A bookmarklet is a small piece of JavaScript code run from an URI. In the same way that the protocol can be specified by the prefix of `http:`, `ftp:` or `file:`, you can also specify `javascript:`. This allows for JavaScripts to be run from the address line or be able to be bookmarked. A JavaScript bookmarked in this way is what has been coined with the term a "bookmarklet".

When a JavaScript URI is executed the result string is loaded as a new page. The JavaScript also has access to the current page's DOM and can make changes to it. If the script evaluates to the JavaScript type of `undefined` it will only operate on the current page and not load a new one.

A very simple bookmarklet to pop-up an alert box on a page would be:

```
javascript:alert("Hello, world");
```

A drawback with using JavaScript in this way is that it is limited to a single line. It is also constrained by how long the URIs are allowed to be in the browser. In Internet Explorer this is 2,083 characters long [1]. Most other browsers tolerate more, but the HTTP/1.1 specification[29] also warns:

> Servers should be cautious about depending on URI lengths above 255 bytes, because some older client or proxy implementations may not properly support these lengths.

To get around these limitations we need to load another script from a file instead of typing the whole script in the URI. This is done by inserting a new script tag into the DOM of the current page and loading a full JavaScript file from that script tag. This is the equivalent to using a <**script src**="example.js"/> tag in HTML to load an external script, but allows the user to call the script at any page they want. A small example of how this is done can be seen in 5.1. The script is expanded for readability, but has to be typed on one line without the comments. The final bookmarklet can be found in the appendix at B.2

The `nocache` part is added to always load an updated version of the script and not one in the browser's cache. The finished code for the bookmarklet also includes a check to see if the element has already been created. The next step was to use the script that the bookmarklet inserts to do something usefully.

5.3 Communicating with the server through dynamic script tags

Originally the plan was simply to use the idea for inserting the script into the page through a bookmarklet and then use the `XMLHttpRequest` object to do asynchronous client-server communication. This however does not work because the security model of browsers disallows cross domain requests with `XMLHttpRequest`. This means that it does not allow communication with servers that are not on the same domain as the server that generated the page. This is a problem for the prototype because it needs to be able

[1] http://support.microsoft.com/kb/q208427/

Listing 5.1: Insert script element from bookmarklet

```
javascript:(
  function(){
    //create the script element
    s=document.createElement('script');
    //set attributes
    s.type='text/javascript';
    // Set the path to file
    s.src='http://example.com/dannotation-js.php?nocache='+Math.
        random();
    //add the element to the DOM
    document.body.appendChild(s);
  }
)();
```

to load on any domain and then send data back to the database running on another domain.

The answer to this was to use the same method as the bookmarklet, by inserting additional dynamic scripts tags into the document. The idea was first encountered by me in an example [30] by Kappe on how to use Yahoo services with dynamic script tags. Normally the desired result of adding a new script tag would be to add an additional external script to the page like in the bookmarklet 5.1. It can however also pass along data from the server as long as the returned data is valid JavaScript(it causes a client error if it is not). When a new script tag is created the browser executes the content returned from the link in the `src` attribute. This link is also used to pass parameters to the server using a `GET` request.

To keep the data as valid JavaScript, everything that the server sends back is enclosed in a JavaScript function call with the returned data as a parameter. The function returned by the server needs to correspond to a callback function defined in the client side script. The data passed as a parameter from the server then gets executed as a parameter to the callback function on the client.

In the prototype JSON[2] is used as the data interchange format between the client and the server. JSON was chosen because unlike XML it does not require extensive parsing with JavaScript. When the JSON string returned from server is used as an input parameter to the callback function, it immediately gets evaluated and added as a JavaScript object. This object can then be accessed like any other object in the DOM. A simple example of how the communication with the server is done can be seen in 5.2. This approach is very close to the JSONP(JSON with Padding) approach detailed in [31], the difference being that predefined callbacks are used in the prototype because it does not need dynamic callback naming. JSONP is also implemented in JQuery, but before that was noticed the dynamic script loader function was already widely used throughout the code.

5.3.1 Weaknesses with using dynamic script tags

There are also several disadvantages to using the dynamic script tag generation approach. One disadvantage of using script elements to exchange data is that there is no

[2] JavaScript Object Notation is a lightweight data-interchange format. It is based on a subset of JavaScript.

Listing 5.2: Server communication with Script tags

```
//Send data to server.
addScript('http://example.com/example.php?firstname=Daniel');

/*
Server returns string 'callBack({"lastName": "Wenner"})'
This calls function "callBack()" on client.
*/

callBack=function(JSON) {
  //alerts "Wenner"
  alert(JSON.lastName);
}
```

way to know if the request is successful or not. Unlike the `XMLHttpRequest` object the script calls only accepts the the HTTP status code `200 OK` and fails silently for all other statuses. Another drawback that is at the same that the main reason for using script tags in this way, is that it does not enforce cross domain security. This means that you have to trust the server that the script contacts to not return malicious data that could harm the client. The returned script runs with the same privileges as the scripts loaded with the original page, so the script can view any data that the user has in relationship with that page (cookies, login names and so on). Therefore running bookmarklets to inject scripts into another page comes with a big risk for the user if the source of the script is not trustworthy. In the case of the prototype this should not be a problem because it is used to load data from a controlled server.

Another scenario is that a specially tailored web-page could attack the code that the prototype injects in it. This could possibly allow the attacking web-page's script to see information sent between the client and the server. Because the system will only be deployed in a very small scale in this project, such specially tailored web pages are not likely to be encountered. It should of course be taken into account if the system is put into production. It might also be a potential problem for systems like Diigo and other systems that inject their own code into a page through a bookmarklet and send information about the user or user actions to the page.

5.4 Positioning highlights through DOM manipulation

The second important part after the user is able to load the system is how to position and store the users annotations. Different approaches were considered for storing the range of text selected and all three methods mentioned in 3.1.2 were an option.

The implementation is this case was intended primarily for educational material and e-books in HTML format, so as high a rate of change in the material as a "normal" web page — such as a blog or news site — is not expected. In the work of Brush et al. [27] they also mention that users still rank the value of orphaned annotations high if the text is changed enough that users understand the need to orphan an annotation.

Because of the assumption that education material will not change very often a robust positioning algorithm has not been given high priority. It is a difficult problem that solving would only give minor benefit in rare cases in the context of this project.

5.4.1 The DOM Range object

The system needs to be able to uniquely find where the annotation was made, both for when it's made and for later visits to the page by the same or another user. This eliminates a pure string search because a string search can have multiple matches in a page. A search for a string also fails if the text spans elements as a text node in HTML only belongs to one element. The string "Hello" will not be found in this example:
<**p**>Hell<**em**>o</**em**>, world</**p**>.
To enable the user to mark elements like that, a system of storing the position of elements within the page and their offset has been chosen. It accomplishes this using the W3C DOM Range object. This allows the script to "cut" out a part of the DOM and store it. The method was inspired by the algorithm behind Brian Donovan's Ahoy system[32]; a system that allows users to mark text on a page and then enables sharing of links to that place in the page. The unfortunate side effect of using the Range object is that it's — as many other DOM level 2 features — not supported by Microsoft Internet Explorer. Internet Explorer has it's own TextRange Object that instead of working on the DOM operates by the storing and X and Y pixel offsets relative to the page. This makes the two methods so different that two completely different code branches would have to be used and two sets of data stored in the database based on the client. Because this would create a whole lot more effort and needless complexity it was decided to not support Internet Explorer.

To make it even more interesting the browsers don't implement the Range object directly, the user selection has to be saved as a Mozilla Selection object, that is not part of the official DOM. The other major competing browsers to Internet Explorer (at least Safari and Opera) also have support for this object, so even though it's not a part of the official DOM, it's still the most "standard" way of getting the user selection. How this is done can be seen in B.3

To the make a Range object out of the selection the function `selection.getRangeAt(0)` is used. This gets the first selection the user made. The getRangeAt() function supports more selections at the same time, but we are only interested in one selection at a time. The last selection the user made is then stored as a global variable while the user decides if they want to add a comment of just highlight the text.

When the user has decided and either added a comment or just saved a highlight, the representation of the range is sent to the database. This is done by the the `annoloader.addHighlight()` function. This function picks out all the data from the global range object that represents the users selection. To be able to place the range in the correct place we need to know where in the DOM it belongs. This is done by finding the parent element in the DOM using
document.getElementsByTagName (parentElementTagName). Then walking through the array and the array's children to find the child that matches the start and end element of the Range object. The algorithm for getting this data is shown in 5.3 The data that is stored in the database to be able to retrieve the annotation later is:

- Tag name of the parent node
- Index of the parent node

Listing 5.3: Getting the data representing the range object

```
//labeled loop, so we can break the specified one
elementsloop:for(var i=0; i< elements.length; i++){
    for(var j=0; j<elements[i].childNodes.length; j++){
      if(elements[i].childNodes[j] == startElement){

        startNodeIndex = j;
        parentIndex = i;

      }

      if(elements[i].childNodes[j] == endElement){
        var endNodeIndex = j;
        //break the first loop when we find the end node
        break elementsloop;
      }
    }
  }
```

- Child index of the Range's starting element
- Character offset from the start of the start element
- Child index of the Range's ending element
- Character offset from the start of the ending element
- The string the user selected. Not needed for the retrieval itself, but can be used to check if the found range is the right one by string comparison.

;

This and the other data about the annotation is then sent to the server. A text marking function(see B.4) is called from the callback function that runs if the appropriate response is sent back from the server. This marking function then recreates the range again from the data received from the server, this should be the same parameters that are sent to the server. When the range is recreated the text is finally highlighted using the ranges `surroundContents()` method that places a <dannotation> tag with some attributes around the text the user wanted to highlight.

5.4.2 Limitations

One limitation that is set on the range the user selects is that it all needs to be within the same parent element. This means that if the selection starts in a one paragraph, it can't flow over into a new paragraph. Selections also can't be over another <dannotation> tag or be blank. The user is alerted in a status field if these conditions occur.

Another drawback to using this method is that the DOM changes when a new `<dannotation>` tag is inserted. This means that the tags have to be inserted into the document in the same order that they were made or the index values could get wrong. The consequence of this is that the user always needs to have an up to date version of the annotations that exist for that page or they could get a conflict if two users make annotations on the same place or the index of an element changes between updates. More on how this is handled can be found in the section about client updates 5.5.7

5.5 Program flow and features

This section gives a description of what the user can do with Dannotation and what happens when they take an action within the system.

5.5.1 Loading the menu

The first thing that happens when the user decides to load Dannotation into a page is that the script is loaded using the method discussed in 5.2. The script then loads the menu and allows the user to log in, or if the user is already logged in it loads the annotations for the current page. A diagram of the process can be seen in figure 11.

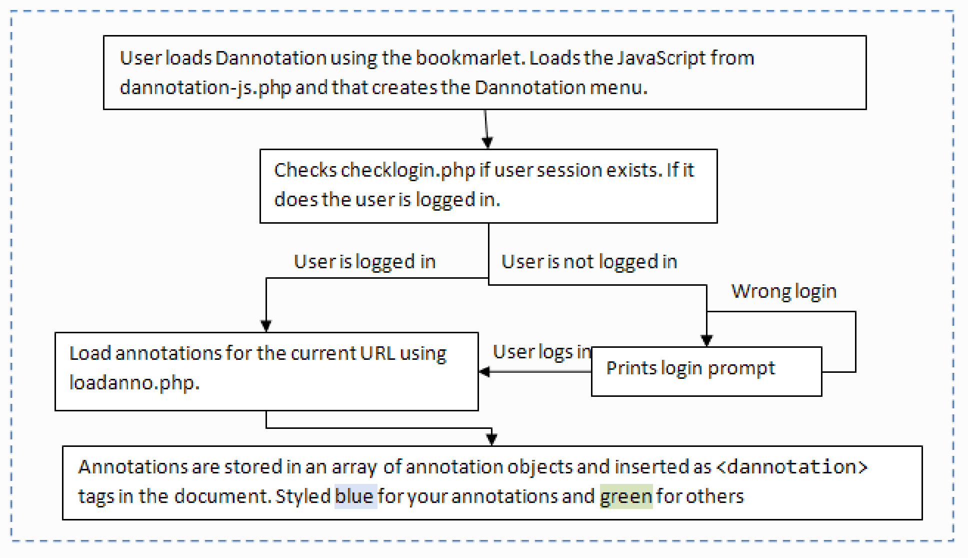

User loads Dannotation using the bookmarlet. Loads the JavaScript from dannotation-js.php and that creates the Dannotation menu.
Checks checklogin.php if user session exists. If it does the user is logged in.
User is logged in
User is not logged in
Wrong login
Load annotations for the current URL using loadanno.php.
User logs in
Prints login prompt
Annotations are stored in an array of annotation objects and inserted as <dannotation> tags in the document. Styled blue for your annotations and green for others

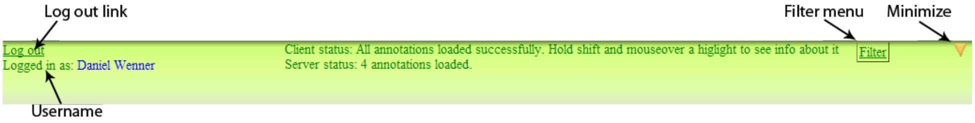
Figure 11: The actions that happen when Dannotation loads

The menu is loaded as a bar at the bottom of the page. This bar is used to give status updates, provide login/logout and filtering annotations. The menu can also be hidden in case it overlaps text the user would like to highlight or otherwise gets in the way. The system still functions if the menu is hidden, but the user won't see the status updates. The menu can be shown again by pressing the maximize button that appears when minimized. The menu is position using a `fixed` position with CSS, so it always stays on the bottom of the screen.

The client status and server status fields changes based on user actions. The client status field is changed when certain actions happen on the client, like a selection, an invalid selection or opening the comments window. The server status field is meant to give updates on the communication with the server, like how many annotations were loaded from server, if it found new annotation on server since last update or if saving an annotation was successful.

5.5.2 Adding annotations

The main purpose of Dannotation is to be able to comment on documents. This can be done when the menu is loaded and the user is logged in. It is done by selecting a piece of

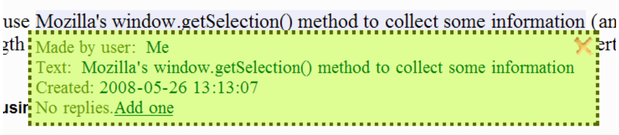


Logged in as: Daniel Wenner

Client status: All annotations loaded successfully. Hold shift and mouseover a higlight to see info about it

Server status: 4 annotations loaded.

Figure 12: The menu bar



use Mozilla's window.getSelection() method to collect some information (an

Made by user: Me

Text: Mozilla's window.getSelection() method to collect some information

Created: 2008-05-26 13:13:07

No replies.Add one

Figure 13: A highlight made by the current user and the information pop-up associated with it

text in the document with the mouse. This pops up a menu with two choices: `Highlight` or `Add comment`.

Adding a highlight acts as a highlight pen that marks the background in the light blue colour used for the current user. This immediately sends the range data to the server without any additional data. A normal highlight and the information pop-up attached to it can be seen in figure 13.

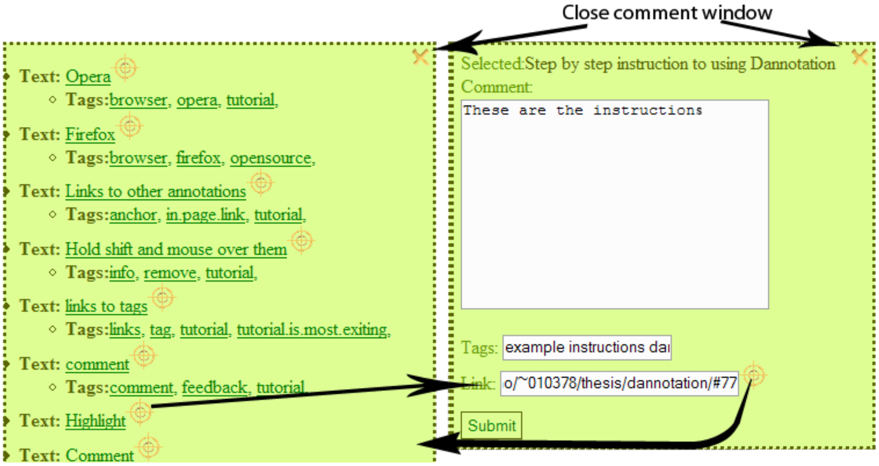
If the user decides to use the `Add comment` function a new box is popped up where the user can type in comments, tags separated by space or a link to related information. The link can also be to another highlight on the page represented by an anchor to the id attribute of the annotation. These can be easily created by pressing the “aim” icon, which changes the comment box to a list of annotations on the page. Pressing the text there jumps to the corresponding annotation and does a short animation of the CSS `font-size` property to make it stand out. When the user presses submit the annotation is saved. The comment and link to annotation window can be seen in figure 14.

When the comment box is created the ability to mark text and save it is turned off, so the text the user marked to bring up the menu is the last marked text saved. It’s this last text that is used to create the range after the user submits the comment window.

When the user is done with their input, either in the firm of just a highlight or after they press submit in the comment box the the parameters representing the currently selected range, the comment, link and tags is sent to the server represented as a URL-encoded string in JSON format. These parameters are then saved to the database on the server to be able to recreate the range later. The same data is also sent back to client after the server script is done saving the data. The callback function on the client uses this returned JSON data and makes a new annotation object that is pushed onto the end of the array of annotation objects in the document. The last thing that happens is that the function to mark the text is called and that function finally puts a <dannotation> tag around the recreated range. <dannotation> tags also include some information about the annotation as attributes to be able to connect the tags to the annotation object array:

ID is set to the same as the annotation’s id in the database.

class is set to either `dannotation_my_higlight` or `dannotation_public_higlight` depending on if it’s the current users annotation or someone else’s.



- Text: Opera
  - Tags:browser, opera, tutorial,
- Text: Firefox
  - Tags:browser, firefox, opensource,
- Text: Links to other annotations
  - Tags:anchor, in.page.link, tutorial,
- Text: Hold shift and mouse over them
  - Tags:info, remove, tutorial,
- Text: links to tags
  - Tags:links, tag, tutorial, tutorial.is.most.exiting,
- Text: comment
  - Tags:comment, feedback, tutorial
- Text: Highlight
- Text: Comment

Selected:Step by step instruction to using Dannotation

Comment:

```
These are the instructions
```

Tags: example instructions dai

Link: o/~010378/thesis/dannotation/#77

Figure 14: The comment and links window. Pressing “aim” icon goes to link mode.

arrayIndex is the index in the array on the client, so the script can find the corresponding annotation in the array of annotations.

username is also stored in <dannotation> for convenience and to give some semantic meaning to the source.

The reason for not creating the highlight on the client immediately, but using the data the server returns is to avoid giving the user the impression that something was saved if it is not. By sending it to the server before marking it, the server can check if something is wrong with the data and return an error.

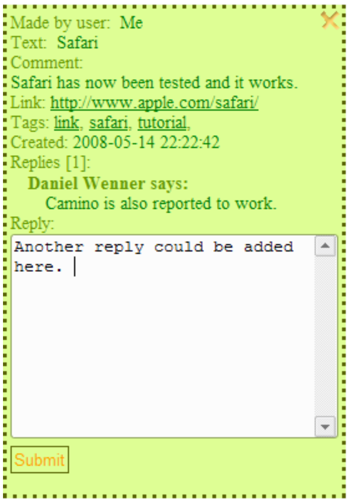
5.5.3 Seeing information about the annotations

To see information about an annotation the user needs to hold the shift button and mouse over the highlighted area for that annotation. There is also a small feedback on normal mouse hover without shift to differentiate an annotation from the background and add some feedback that this is not just text.

The information pop-up that is created contains: The username of the user that made it, the text that was marked, the date and time it was made, a comment, tags, the link and can contain replies from others or the same user. The comment, tags and link only shows up if they contain something, so a pure highlight only shows username, highlight, text, date and optionally replies(There is also a bug that shows tags even if there are none if the user just made the annotation and has not reloaded the page).

5.5.4 Adding discussions to annotations

When someone has made an annotation either with comments or without it can be replied to. This is done by clicking `Add reply` after bringing up the information pop-up for an annotation. This brings down a form to fill inn the comment and a submit button. When the submit button is pressed the text and id of the annotation is sent back to to the server and the reply is saved for that annotation. This also closes the information pop-up box to avoid problems with the sequence of replies because of asynchronous communication. The user is given a message that they need to open the information pop-up again to see their reply. An information pop-up with replies looks like figure 15 when



Made by user: Me
Text: Safari
Comment:
Safari has now been tested and it works.
Link: http://www.apple.com/safari/
Tags: link, safari, tutorial,
Created: 2008-05-14 22:22:42
Replies [1]:
Daniel Wenner says:
Camino is also reported to work.
Reply:
Another reply could be added here.

Figure 15: The information box with reply

the reply form is open.

The list of replies is a simple list of replies based on the sequence they were made in and the user that made them. Because of limited space in the information pop-ups and a desire to keep down the amount of screen real estate, a threaded system or other such discussion enhancing features has not been implemented.

5.5.5 Filtering

To the right side of the menu the filter menu can be found. Pressing this button brings up a sub menu with the filters. It only does filtering on the client side, and just toggles the visibility of the annotations off. The user can still see the border that appears if the annotation is moused over and open it if he holds shift. It is meant to make reading the text easer if the number of annotations get so high that it disturbs reading.

Currently the only supported filtering method is by users. It can filter by your own annotations, all annotations belonging to others or just by a specific user. The font size of the elements toggled are animated to go to a larger size before returning to the original size. This is done to make it easy to see which annotations are filtered because it was hard to see before this feedback was added.

5.5.6 Pup-up dragging and moving

Because the information box and the box to write comments in would sometimes get placed over text the user might need to read the ability to drag them around was added. This is implemented using a plugin to JQuery called ppDrag.[3] A downside to using that plugin is that custom mouse down events had to written for the forms contained in the dragable boxes because the text marker could not be put there using the mouse. The function to fix it turns off ppDrag when the user presses the form and turns it on again when the user releases the mouse button. This lets the user put the text marker into the form fields and still enables drag on the rest of the box.

When the user wants to add a comment to something or make an information pop-up

[3] ppDrag is a Drag&Drop plugin for jQuery, which mimics the interface of jQuery UI's Draggable, but is focused on performance.

appear the pop-ups are created near the users mouse cursor. The X and Y coordinates where the triggering event happens are sent as a parameters to the function creating the pop-ups using `e.pageX` and `e.pageY`. When the user uses the “link to other annotations box” this box is also moved close to the location of the anchor the user presses, so the user does not loose the box when they navigate to an anchor in the page.

5.5.7 Keeping annotations up to date

Because users could potentially add annotations at the same time without annotations being fetched from server, this could create an orphaned annotation if they overlap or the DOM changes enough to give it the wrong position. Therefore server is checked on each mouse up and when transmitting annotations to see if there are additional annotations on server before adding the new one. If new annotations are found they are dynamically added to the document as they are made by other users. The source for the client update can be found in B.5.

This can create a race condition because there is no way to know if the script executed successfully or not. If one annotation is added in the same spot as another user just saved an annotation that has not been updated to the first users client, and that client manages to send the annotation before getting the update from the server both users could mark the same spot and possibly create problems. This possibility is however very small under normal use, but could be used to exploit the system.

5.6 Robustness

5.6.1 Security

The Dannotation system has been created as a prototype to be used for examining features and testing on some users and is therefore not very robust to deliberate attempts to break it. User submitted data is not checked extensively and the possibility exists to insert data into the database without using the Dannotation client by being logged into the system and submitting variables through an URL. As explained in 5.3.1 it might also be vulnerable to cross-site scripting attacks.

Too much trust is generally put on the client to decide where annotations should be placed as there is no check from the server about the page content. A client that lies about these positions or transmits other faulty data might be able to fool the server to accept bad data.

5.6.2 Performance

Dannotation does perform reasonably well. The Dannotation JavaScript is 104KBs, so it does not take long to load on a broadband connection. It has not been tested with very low bandwidth like a modem or ISDN line, but should work there too with about 14-15 seconds of patience.

It has also not been tested on very low end PC’s, on all the machines tested actions happen fast enough to not have a notable delay. The exception to this is the loading of annotations, it can take be a noticeable delay if there are many annotations on a page. Why this happens can be described in 5.7.3.

One area where problems are foreseen are with massive amount of users. Like many web 2.0 solutions dependant on asynchronous communication a lot of request are made to the server. Especially the update function might need a reworking as that happens on each mouse click currently. Other systems like Diigo, Sharedcopy and Trailfire are also plagued by slow response times, so performance is a general problem among the available solutions. Without having tested with many users, at least at the moment Danotation is faster than many other systems in use.

5.6.3 Pages it does not work for

The Dannotation system can break on pages that contain jQuery or other libaries that uses the $ as a shorthand variable for functins. Except trying to keep a naming convention on functions, classes and ids not many steps are take to ensure that it it works perfectly on all pages. The original pages CSS can e.g. cause Dannotation to look different depending on the page. Other JavaScript on the page can also interfere with the operation of Dannotation.

5.6.4 Spam

Something that plagues many services on the internet where users can communicate with each other is spam. Dannotation currently has no features to combat potential spam, but things like voting, reporting on users or CAPTCHAs could be a part of combating it if it ever became a problem for annotation services.

5.7 Problems and workarounds

5.7.1 Highlighting text

One problem with the system is that it does not work on text included directly in the HTML `<body>` or `<div>`. There is a specified list of tags that the `onMouseUp` event fires on and those two tags have been omitted from that list. The reason it does not execute in those situations is that the menu is a child of the `<body>` and is contained in a `<div>`, so when they are included in the mouse up event it's possible to highlight parts of the menu. As this is something that is not desirable selection is disabled on those two tags. Some workarounds to this has been tried, but none worked satisfactory.

Another slight problem is that the system does not detect if the user highlighting is over another highlight immediately. The check for this happens right before sending the annotation, so the user does not get informed about the invalid selection before they want to save the highlight. If they add a comment this does not happen before after the comment is entered, so the user could loose the carefully written comment. This can also happen in conjecture with the updating of annotations, so if someone else adds a comment to the same place while the first user is writing the comment and then presses the save button he will get an error message about not being able to mark over other annotations. This works as intended, but is not a good solution from a user friendly standpoint.

5.7.2 Transmission size

The size of the data transmitted to server is limited to 2000 characters in the script to avoid problems with sending the `GET` request. This means that how much the user can write in the comments and the amount of text selected is limited. Even though most modern browsers support more than this Internet Explorer(2,083 characters) and web servers don't[4]necessarily do, so 2000 is set as a conservative number to maximize browser compabillity and to set a limit, so users don't transmit massive mounts of data.

[4] A comparison between different supports can be found at `http://www.boutell.com/newfaq/misc/urllength.html`

5.7.3 Rebinding events

Because Dannotation frequently creates new HTML elements these new elements also need to be bound to the events that should happen with them. When using JQuery to bind the new events the selectors for binding new events would bind the same event several times to other elements in the document. This caused things like calling the function to open the options after selecting text to be called multiple times and would take some time.

To work around this all bindings in the documents are removed before each new `<dannotation>` tag is inserted in the document and reapplied when the insertion is done. This removing and reapplication of event bindings is moderately resource expensive and not very elegant, but does keep the event triggers bound correctly. It gets more expensive the more annotations there are on a page, but has not yet been seen to cause a noticeable slowdown in normal use. It does however add some time to loading of many annotations when the system loads because it rebinds for each annotation added.

5.7.4 Sharing

Because of the way annotation are stored they are dependant on the order of insertion for a given page. Inserting a highlight changes the DOM of a page and the data stored about a highlight is the position in the DOM, so correct insertion order must be kept across all users. This is also why a solution to update annotations like the one described in 5.5.7 was needed.

A solution to keep private and shared annotations from interfering with each other was not found. A situation could easily arise where one users private annotation would overlap someone else's shared annotation and the user with the private annotation could get an orphaned annotation. Because of this problem and the desire to investigate how users react to shared annotations it was decided to make all annotations shared.

6 User testing of prototype

6.1 Test set up

The test with users where done in two parts. One part was done with first year master students who were going to use the system during parts of their course for “IMT4491 Databases in electronic publishing” at HiG. The other group consisted of first year technology students taking the courses “IMT1291 Web design” and “IMT2491 Web publishing” also at HiG.

6.1.1 First group

The book the first group were going to annotate was “Introduction to Information Retrieval” [33]. This book is available both as PDF files and in a HTML version exported with LaTeX2HTML and the hope was that the HTML version would be used to make use of Dannotation. As these students all have a pretty high level of interest and education in web development it was felt that their comment would have a high degree of validity.

Dannotation was introduced for them through a session of about 10 minutes out of a lecture where use of the system was demonstrated and the use of it encouraged when they were reading the book. An email F.1 to the whole class was also sent after the meeting with them. At the point the prototype was ready for use the class only had one week of three weeks of lectures left that pertained to the subject of information retrieval. There was still a bit over a months time to use the system to read for the exam in the subject.

Right after their exam in the subject was finished the class was interviewed on a one and one basis. Three students were interviewed, and a fourth asked one question. The conversations were recorded on a Dictaphone as Windows Media Audio files. They were asked question about what they thought of using the system and on what they thought the potential for such systems were. As two of the participants are also teachers they were asked if and how they would consider using a system such as Dannotation in their own classroom. Some follow up questions were asked in some cases to get additional information.

6.1.2 Second group

The second group were primarily instructed to annotate on tutorials w3schools as this was a part of their resources for the course. It was also suggested to the web publishing group that they could use it for other reference sites such as `php.net`. The system was introduced to the students on the 8th. of May and they could use it until 2nd. of June when the web design class has their exam. The introduction was done in a one hour lecture about JavaScript that was held by me, where the system and how to use it was presented at the beginning of the lecture and also a plea to use it at the end of the lecture. Seven people out of a reported 41 potential students showed up for that lecture, so an email F.2 was also sent out containing information about the project and that help was needed. A message was also put up in Fronter[1] the 14th. of may including the same

[1] Fronter is a Learning Management System used by HiG

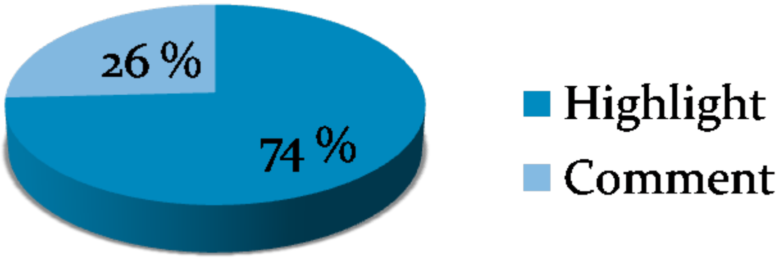


Figure 16: Type of comments made by users

instructions.

A questionnaire to the students was planed, but was not given out due to the fact that the low number of participants are not enough as a base for statistical evaluation.

6.2 Test results

6.2.1 Annotations

The test results are somewhat limited as most users did not make that many annotations. There were only six users that registered for the system that were part of the testing group. Out of these only four made any annotations.

The user that made the most annotations made 23, all of them in the information retrieval book. The user that came second made five annotations that were all on the page[2] describing Dannotation. The same holds true for the two others that made two and one annotation each, they did not use the system outside that homepage. How the number of annotations were divided can also be seen in figure 17

Eight out of 31 total annotations were comments(25,8%), the rest were highlights as shown in figure 16. The user that used it in the curriculum made three out of four comments that were all somewhat about the text commented. The last one was the comment “Institutional comment test” after marking the word “Institutional” . The rest of the users contributed these comments: “hei på deg”, “TEST”, “test” and “asgafdfda”. A table with all the results can be found in 19. As these results show, most of the users did not make it through any more than basic testing of the system on the page it was presented, so litte can be gained from them.

6.2.2 Interviews

As mentioned three students were interviewed after their exam in the subject they had the opportunity to use Dannotation on. A fourth student was also asked why the system was not used, with the answer that “It seemed complicated[When the system was shown in the class]. Did not feel like I could contribute”.

[2] http://stud.hig.no/~010378/thesis/dannotation

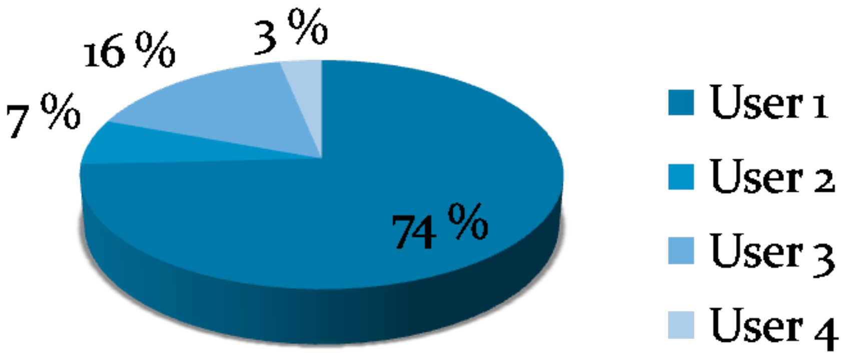


Figure 17: Percent of annotations made by users

A list of questions was made up beforehand, but not all questions were asked to all users because of different levels of use of the system and their background. Some additional follow up questions were also asked in some situations. The answers are presented by user and as a summary of their answers. A rough list of the base for questions can be found in appendix E.

First student

The first student used Dannotation to make one test annotation on the start page, and had also been given a small introduction to it individually because he was not present for the classroom introduction. The student has a background of web development and said he also considers doing his master thesis on new methods to use learning systems for distance learning and cooperation. He said that he though Dannotation could be helpful in such an environment because it shares notes with others. He further notes that it could be helpful for teacher/student communication, student/student communication and for distance learning.

Other that that he liked the solution with the bookmarklet and thought it was a good solution to load the system. As a comment on the features of Dannotation he liked that it worked, but had no further comment on what he would like in an annotation system or what was missing from the system.

Second student

The second user was the user that made 23 annotations. He He also has a interest in web development and works as a teacher.

About the usefulness of the system he commented that it would have been useful with more users. He liked the concept of sharing information you think is important. He thought it might make the reading process more effective if many people are commenting and marking parts of the text.

On a question about how he used it he answered that he marked what he thought was relevant and important parts of the text. He also thought bookmarklet was an good solution and he did remember to use it when he wanted to highlight something. He had some difficulties using the system however because sometimes text could not be highlighted and it did no give a message about why(because it's directly in the <**body**> tag as explained in 5.7.1).

On functions he missed from the system he mentions that he wanted an easy way to look at comments that were made and get the information others had added. This exists in the system by holding shift and mousing over the highlight, but this user did not discover that. As seen in 6.2.3 this was also a more general problem with the system.

On the question about added learning he answered that it could be helpful to make the reading more effective because you can see where the important elements are in a text. He also agreed that it could be helpful if the teacher made some comments in the text. He thought it could be useful for corporation with with a more expanded module for comments and relating it to other information.

As for disturbing the reading when there was annotations resent, he did not feel this was an issue, but the fact that marking text sometimes failed was annoying. Compared to annotating a normal book he was pleased with the added possibilities Dannotation gives and did not feel any pressure about the fact that annotations are shared.

He also agreed that it was a tool he could use himself as a teacher, but also wanted additional features to relate tags and comments to related sources.

Third student

The third student did not use the system himself, but did see it used. He is also a teacher.

As he is always looking for good additional resources while reading he thinks such a system can be useful to find the important things to read. As a teacher he think his students would be very happy with a system that lets them be able to find the important information easily because they as he said "are interested in reading as little as possible". As a tool to increase learning he thought it would help to find just the parts that are needed in a text, but that it is a short term learning effect. Compared to annotating on paper he thought it would be about the same, with the added benefit that the annotations stay there.

A feature he would have wanted is also to navigate between the annotations using only the keyboard.

6.2.3 Views of annotations

Because of low response from the users to the annotation system, there was also added a function to report back to the database when a user opens an annotation to view the comment. This records the user id, the annotation that is viewed, the current time and the users IP address to see if they are installing the bookmark in several locations or accessing it from the same network each time.

As the user with the most annotation said in the interviews he did not know that you needed to hold the shift button and mouse over the annotations to see the information pop-up. After the message to use the system was written in the Fronter room the 15th of may only one of he users looked at any annotation, but did not make any after this period. One user in addition to the most active user made annotations in the period after the function to count views was put in place, but neither of them looked at their own or other annotations.

This suggests that the requirement or explanation that the shift button needs to be held while mousing over the highlight was not good enough.

7 Discussion

7.1 Prototype

7.1.1 Features

While the prototype lacks some features it does satisfy most of the demands that were set for it in the system design section 4.3. It is in many ways a more lightweight solution than some of the other available annotations systems, and this allows for making quick highlight or comments without taking too much concentration away from the reading. There is also no need for a plug-in or other additional software to load the system and it works in all the newer tested standard compliant browsers. Unfortunately it does not work in Internet Explorer because of different handling of the Range object from other browsers.

It can also be loaded on any page and has relatively minor restrictions on what elements it can not mark. When this is compared with systems that implement a proxy solution where you most often have to browse the pages through their pages or the systems where documents have to be uploaded to their server and can only be used there, this solution provides added flexibility. The downside of this solution is of course that you have to press the bookmark on every page the system should load on and this can be easily forgotten, especially if you only want to view other peoples annotation for a page. In the interview rounds all three master students think that pressing a bookmark to load the system was an all right solution. If you can get the users to install a browser toolbar this will in most cases beat a bookmarklet for functionality simply because they integrate into the browser and can run with the browsers privileges. Extra toolbars can however quickly bloat the layout of the browser, so care must be taken to not alienate the users by included unneeded features or provide easy configuration to be able to select features.

In the Dannotation prototype there is no function for users to configure options. Adding the opportunity for users to configure some behaviours like the colour of annotations could have made it more user friendly as the highlight colours can be hard to see on some monitors. It was also considered to change the relatively light colour for the highlight, but it was not done because of a feeling that it would take more focus away from the text.

The option to delete or edit comments is also something that should have been included in the prototype. Editing the comment is easy to implement, but deleting the highlight is a harder problem because the annotations are dependant on the order they are inserted in as described in 5.7.4. When one annotation is removed other's might need to have their positions updated to remain in the correct spot. This is a consequence of the method used to store annotations, and is something that could have been avoided with a better way of storing annotation positions.

A “homepage” for each user where they can see an overview of their own annotations and contributions should also be something included in a production system. Dannotation lacks any such system except for the system to view other things tagged with the same tags. An example with a feature rich such solution is Diigo, who has excellent search and organizing tools for annotations made and also create a social network around the annotations by suggesting other users that have seem to have similar interests and other pages about similar topics.

Out of the planed features from the system design the ones that did not make it into the prototype before user testing were: voting, sharing to groups and preservation of context.

The voting function for annotations was planned, but was unfortunately not implemented in time for the user tests. The plan was to implement it as a change in colour saturation with those highlights that had a high rank getting darker and more noticeable colours.

Sharing to certain groups was not implemented because of the decision to have all annotations shared with everyone. The same problem would occur with groups that would happen with private annotations like described in 5.7.4. If one annotation was shared to a group and someone not in that group made an annotation in the same spot, this could create a potential orphaned annotation for members of the group that view that page as they would get overlapping annotations.

Storing text close to the original text to preserve context was also one of the planed features. It was cut because of limitations on how much data could be sent back to the server in a GET request as described in 5.7.2. Another reason for cutting it was that a search system – that would have been the primary use for showing the context – was not developed.

7.1.2 Comparison with other systems

For comparing Dannotation with other systems it was chosen to use Glover et al's., comparison matrix [16] as shown in 2.2. As more features than are mentioned by Glover have been discussed in the wanted features section 4.3, the matrix has been updated to also include the some of the features that the were picked out after the problem analysis. This gives a new list of conceptual and technical solutions where new additions to the matrix are marked with a "*". For eLaws some features may not be correct as some are deducted from the article describing the system and not from actively running it. Some of the systems described in the original matrix have also bee cut because they do not exist anymore more such as third Third Voice or they did not run properly when tried, such as Gibeo.

The essential conceptual factors

a) *Text annotations*. Textual annotations are important to allow the annotators own comments to be added at a specific place in the document. This adds additional context to the text and is important to give additional learning benefit.

b) *Private annotations*. Important to allow users an alternative to sharing. Users might have reservations about posting a comment if others can see it and private and shared comments often have a different purpose as shown by Marshall's study [2].

c) *Shared annotations*. Important to get the views of others about the document and what they think is important. Must also be present to give collaborative learning benefits to the annotation system, it is moved to the essential concepts in this matrix.

d) *Login system**. Is essential to provide accountability to the annotator and to give a way to measure user credibility in a shared annotation environment.

The essential technical factors

e) *No additional software necessary*. To be able to use the system without installing additional add-ons or programs is important to lower the usage barrier.

f) *Accessible*. The system should be able to be used with the help of screen readers and be able to be navigated by the keyboard.

g) *Browser independent*. It should not matter what browser the user is using. The system should be able to run in the majority of modern browsers to maximize the amount of potential users.

h) *Maintains page integrity*. The system should be about the same when viewing with the annotation system as it would be without it. This is too ensure that the original content is still the important part.

i) *No contact with hosting server**. A system that does not store or need to download anything from the server hosting the content being annotated has less chance of potential copyright problems. It also cuts down on the amount of processing the server has to do.

The desirable conceptual factors

a) *Graphical annotations*. Adding various forms of graphical annotations lets students annotate with more imagination.

b) *Linking annotations.* The ability to link to related material not provided in the text gives more value to the text. It can also create a new hyperlink structure for the annotated material.

c) *Annotation reports*. The ability to create structured reports of annotations allows them to be found later and allows them to presented outside the confines of the original context.

d) *Sharing to groups*. Allowing sharing to a specified group gives the opportunity to only reach the intended audience for an annotation and gives the opportunity for a smaller group to cooperate.

e) *Voting**. Enabling voting or similar user feedback systems gives the system a way to rank annotation based on user input. This enables the ranking of annotations after perceived use and can be used for filtering or visualising annotations differently.

f) *Tagging** Has become standard way to categorize content in web 2.0 systems. Enables users to categorize and retrieve annotations in an easy way.

g) *Searching**. Useful for retrieving annotations again.

h) *Filtering**. To filter annotations is useful to find just the category of annotation wanted.

i) *Discussions**. Enabling this on annotations provides a way to get additional opinions on an annotation and can also pride a way of discrediting false information.

The desirable technical factors

j) *Open architecture*. Having an architecture that allows developers to add to the system is helpfully to be able to adapt the system for different people.

k) *Non-commercial*. An advantage in that it does not place a cost on the user to use it. Also more likely to be an open architecture.

l) *Preserve context**. Eases retrieval of annotation later as it is easier to remember the annotations with the context showed.

	Essential									**Desirable**											
	Conceptual				**Technical**					**Conceptual**									**Technical**		
	a	b	c	d	e	f	g	h	i	a	b	c	d	e	f	g	h	i	j	k	l
Annotea	x	x	x	x	p	p	p	x	x	x	-	-	-	-	-	p	-	-	x	x	-
Diigo	x	x	x	x	x	-	x	x	p	-	p	x	x	p	x	x	x	x	-	-	p
eLaws	x	x	x	x	x	x	p	x	-	x	x	x	-	-	-	-	-	-	x	x	?
Shiftspace	x	-	x	x	-	-	-	-	x	x	p	x	-	-	-	-	x	p	x	x	p
Processed book	x	x	x	x	x	p	x	p	-	-	x	x	x	-	-	x	-	-	x	x	p
Sharedcopy	x	x	x	x	x	-	x	p	x	x	p	x	-	-	-	x	-	x	p	-	p
Co-ment	x	x	x	x	x	-	x	p	-	-	-	x	p	-	x	x	x	x	x	x	-
Trailfire	x	x	x	x	p	-	-	p	?	-	x	x	-	p	x	x	-	x	-	-	-
Dannotation	x	-	x	x	x	-	p	x	x	-	x	p	-	-	x	-	x	x	p	x	p

Legend: x, supported; -, unsupported; p, partial support,; ?, unknown

Table 1: Comparison matrix between different annotation systems

From the table it can be seen that Dannotation feature list seems to compare to the established systems reasonably well, even if it does miss a few of the essential factors. Like mentioned in the features section7.1.1 it lacks private annotations, sharing to groups and the planned voting and search interface. It is however more lightweight and performs faster than most of the other systems that can feel very slow to use. It is also focused on not cluttering the page unlike Sharedcopy, fleck, Shiftspace and some of the other systems where annotations become a more major part of the page than necessary and don't really maintain the integrity of the page and take focus away from the reading.

It can also be noted that no systems tried or found in the literature review really support a direct way to supply votes to annotations. TafAnnote [19] explores the possibility to weight the annotation based on consensus within the discussion for the annotation, but not for a simple like/dislike dialogue. Trailfire provides a voting system like this, but only for Trails, and those are collections of sites and not one comment. Diigo has ranking and counting systems for tags, bookmarks, viewers of a page and other usage statistics that while the users don't give a active yes or no they can be said to vote with their general use of the system like in other social bookmark sites. It would be interesting to see a system that incorporates a direct user feedback system for only annotations to see if it can provide a useful measure to filter, sort or visualise annotation by. It can also provide a way to filter out "false authority" where the contents of an annotation can change the perception of the original text and provide incorrect information.

Another area that seems to need more focus in annotation systems is the subject of accessibility. According to the eLaws article it is accessible, but all other system tried would not work properly with something like a screen reader. Annotea gets a mark for partial support because clients could be created that support accessible solutions. The processed book also gets a mark for partial support because it inserts it's marginalia directly in the page with alternative text that displays the title of the annotation and could be read by a screen reader. A standard way to create annotations would be helpful to be able to support user in need of accessibility options.

On area where Dannotation lacks some work is in the area of annotation reports. The only way implemented currently that supplies a list of annotations is in the tag interface where you can see other annotations tagged the same, This is connected with the fact that there is no "homepage" for each user in Dannotation which would be a natural place to supply them with a view of their previous annotations.

7.1.3 Learning benefit compared with other systems

The most important feature to achieve a cooperative learning experience is the feature to include sharing of annotations in the system. The focus in this thesis has been on other systems that also support sharing, so all the systems in the comparison matrix support shared annotations.

To achieve any learning benefit it is of course important that the students actually use the system. As the use of Dannotation was limited conclusions about how much it would help learning is based on the evaluation of the features and usability of the system.

Because Dannotation does not take up much place in the reading view it should not distract students from the activity of reading. It is also loaded into the content itself instead of forcing the reader to read with it through the systems web page like the Processed book, eLaws, Co-ment and partly Sharedcopy, this should feel like less forcing to use the system than if this was the case. This can of course also make it harder to remember to use it as the user has to press the bookmarklet to load the system. Diigo and Trailfire's Firefox sidebars and browser integration is better suited for this task as annotations are always visible as long as the extension is activated.

Using an annotation system can also motivate students to read and study in a more consistent manner as shown in [17]. This might be attributed to the fact that your involvement with the text increases when you make a comment to it. The student might want to check up on the comment to see if someone agrees or disagrees with it and in that way become more familiar with the material. Dannotation supports discussions on annotations so this would also be a possibility for students to use there. It could maybe benefit from a notification system however to support readers who wants to know when their annotations have been replied too.

As Dannotation forces users to share their annotations the hope with this decision was that it would make more thought out annotations with communicative intent as suggested by C. Marshall in [2]. This has the downside of making Dannotation unsuited as a personal annotation service. Private annotations are also one of Glovers essential qualities [16] for an education system to get users to contribute. In Nokolainen et al's. EDUCOSM study [12] it was also suggested that users get more use out of their own annotation than those of others. While this choice was probably not the reason for low participation a system for learning ought to have the choice between shared and private annotations.

7.2 User results discussion

Because of low participation the planed questionnaire to students using Dannotation was not handed out, as it would not have given any meaningful quantitative results. The same applies for analysing the results of the students annotation behaviour. This looses a big part of the planed results from the user participation, so the research question about the utility and learning benefit of the system had to be answered by qualitative evaluation of the system and results from the interviews.

7.2.1 Interviews

While not all the students interviewed used the system a lot they all have some insight into web development and as two of them are teachers they have experience with teaching others which should make the results from the interview more valid than if they were random students.

All of them were positive to the concept of a shared annotations system. They all agree that the sharing of annotations can make it easier to find related material and that this can make reading more effective, so there is at least a psychological belief that annotations are useful. This also matches the results found by Nokelainen et al. in [12]. It could also help with distance learning as it could become tool for teachers and stutendts to keep contact and provide in context questions or help directly in the material as sugested by the first student. A possible thing to study for further work on shared annotation systems could be to study the effect of shared annotations in a distance learning environment like HiG is starting for media technology master students the fall of 2008.

Another notable thing about the interview with the second student was that he had not found the functions to view comments and the content of annotations after they were made and therefore suggested a method for discussions on annotations that is already implemented. This shows at least partially that some of the features in Dannotation are also something wanted by users, and it is one of the new factors added to the comparison matrix in 7.1.2. The third student also wanted more accessibility, so it could be used with only the keyboard. This is also one of Glower's essential factors that is missing from Dannotation.

The third student also looked at Dannotation as a way for his students to quickly find the important parts of a resource based on fellow students comments. Quickly finding the relevant pieces of information is what information retrieval is about and this suggest that the new structure in the document created by annotations can also be used for that purposes with more weight being put to marked sections of the document. A document with high annotation activity can also be considered to be more interesting similar to how Google's PageRank measures a sites usefulness by how many people link to it.

8 Conclusion

This thesis presents a system called Dannotation that aims to be a easy to use and useful utility for students reading educational material online. This system was used as a basis to answer the questions about how useful a shared annotation system can be in a learning environment. The conclusions are presented as answers to the four research sub questions.

8.1 What attributes are important for a functioning annotation system to support and extend reading and understating of electronic reading material?

In the first question different solutions that are included in other annotation systems or present in related literature have been found. The problem analysis was also used to find some new attributes that are not present in most systems that could be a useful addition to annotation system to ease retrieval of only the annotations the user are interested in. An extended comparison matrix based on the work of Glover et. al was created with these aditional features. The features that I have evaluated as the most useful from the comparison matrix in 7.1.2 are listed in this conclusion, but the rest are also things that should be a part of a very comprehensive system. From this I have found that the attributes most important to include in an annotations systems to support reading for comprehension are::

8.1.1 Textual annotations

To be able to express meaning in an annotation to themselves or fellow learners it's important to include a system that allows for the adding of explanatory text in situ. This allows the annotation to build on the original text as the context of the annotation and allows later readers — which can be the annotator or a fellow student — to get additional information about the text.

8.1.2 Sharing annotations

Allowing the users of an annotation system to share their annotations with each other enables them to: collaborate, provide additional information to each other, ask questions of their peers, allows the teacher to add comments to the text and gives an extra incentive to study the text to make meaningful annotations that others can appreciate.

The Dannotation prototype shares all annotations with all users. While it's also important to allow private annotations sharing everything potentially creates more additional value to the text because all annotations made benefit everyone. The downside of this again is that it potentially creates a distraction from the text itself with many annotations. A filtering system based on votes, tags or other criteria can be used to minimize this.

8.1.3 Log in system

Keeping control of the identity of the annotator is important to provide a measure of trust in the person making the annotation. While it is a slight hassle for users to log in it also makes it slightly harder for users to create negative contributions such as spam. Users could also be assigned roles such as “teacher”,“student” or even“author” to provide a measure of the users authority. Authority could also be dealt with using a rating system for users based on votes for their annotations.

8.1.4 Maintain page integrity

The system for annotating the content should never take focus away from the main task of reading it, but should be a way to get additional information about something and support the reading activity. The system should also not have the ability to change the display of the original content within the system, such as e.g. Shiftspace has as this can lead to confusion over what the original content should be. Ideally the system should be able to be used on the original page and not be required to be somehow imported to the annotation system.

8.1.5 Discussions

In addition to provide textual annotations at a specific place in the document it is important to open up for discussion around those annotations. This allows students to comment each other’s annotations, correct incorrect information, provide more information, answer questions or discuss the learning material.

8.2 How does shared annotations affect the reading and involvement with learning material?

This question was planed to be answered by the user test, but as they did not provide much data the interviews, related work and the problem analysis have provided the data foundation to answer the question qualitatively.

From the interviews it can be concluded that they are all positive to a shared annotation system. This is backed up by other studies where a high percentage of users are positive to using annotation systems in education(87% in [16] and 100% agree or better in [12]). On the question if they think it will improve learning the intervied students they all think that it would would at least make reading more effective as one can find the important parts of a text more easely. This has also been found in other studies, so students have at least a psycological belief that taking notes and vieving other peoples notes help their own learning.

Students taking annotations have also been found to read more actively in [13] and in [12]. In the Wang et al. [17] study the students using the annotation system also read more in the beginning, but it did not show in the final grade. All studies and the work in this studies point n the direction that sharing annotations is useful for student involvement with the text, as they are motivated to read more consistently. Further work is still needed on the actual effect of this and if peer to peer sharing of annotations is helpful to the students on more than a psychological level.

8.3 What technical challenges exist when implementing a system for sharing annotations?

The prototype development and looking at other systems has been used in order to answer the third research question about what the technical challenges for an annotation system are. In the Dannotation prototype and in other web annotation systems the main technical challenges are:

- How to load the system into the content or with a proxy solution how to load the content into the system.
- How to store, retrieve and position the annotations.
- How to handle the shared annotations.

Other technical challenges like keeping track of the the asynchronous communication, handling users, event handling, how to scale the system with a massive amount of users and DOM manipulation troubles were also encountered, but these are more general web development problems than specific to an annotation system.

8.3.1 Loading the system

The way to load the system into the page with a bookmarklet is a standard way to load systems into 3. party web pages without the use of plugins. By selecting this solution one does not have to handle the document itself like the solutions that load documents into or through the system, except by handling the pages DOM in JavaScript. Combined with the cross domain communication provided through script tags this seems to be the most sensible way to provide an annotations system without resorting to browser plugins or keeping the user surfing through the systems web page as many other systems resort to.

8.3.2 Positioning and storing the annotations

The positioning scheme used borrows the basic idea from from Ahoy[32], but unlike that solution the results are stored in a database and used to rebuild the same DOM for later page views and not just to create one link with one highlight. While this method also causes some problems for other things like sharing to different groups, deletion and some limitation on what users can select; it does manage to mark and retrieve the highlight near 100% of the time. The problem of how to robustly position annotations after a slight page change is still a problem however as annotations can quickly become orhaned.

8.3.3 Sharing

The architecture for handling the sharing of annotations had to take into account that users could be making annotations at the same time and that these could interfere with each other. To solve this a system to check for updates from others before an annotation is placed was created that minimizes this problem. This system updates the page the viewer is looking at asynchronously on each mouse click and loads annotations as others make them. No system that does such dynamic updates to annotations has been found, so as far as I know Dannotation is the first system that solves the problem in this way. A notable system like Diigo seems to not handle the fact that annotations can happen at the same spot at the same time and give unpredictable results when this is tried.

8.4 How can annotations generate a richer structure for electronic documents and how can this be utilized?

This question is answered through the features developed in Dannotation and by looking at related work.

Annotations can be considered to mark important or interesting parts of the text in most situations. The kind of links that Dannotation provide between annotations generate a new hyperlink structure with new anchors on pages they are used on. Dannotation also provides outbound links that creates a link to related material from the current place in the document.

A potential use of such information is for use in information retrieval purposes and for user profiling. In information retrieval it would be used to provide additional information about a document an could give hits on pages based on it's annotations. The number of annotations on a page could also be used to evaluate how useful it is as a page with many annotations will likely be a more interesting page.

Users marking interesting passages is also something that could be used to build a profile on users. This should provide more accurate data about user interests than just looking at page views for advertisers as users are likely to be interested in what they mark. It can also be used as a suggester system within the system to find users that are likely to have similar tastes and interests.

9 Further work

9.1 Effect on users

No studies have so far been able to come to conclusive results about how sharing annotations affects users or if it is helpful for increasing the amount of learning that is done. As no conclusive results were reached in this study either, this is still an area that needs further research. Employing a shared annotation system like Diigo for several courses and dividing users into groups that don't annotate vs. a groups that do and comparing results through the achieved grades and by gathering other data from the participants is one way to do such a test.

9.2 Further testing and development of the prototype

Dannotation does contain most of the essential factors for a shared annotation learning system. Working on fixing some of the things it does not do well, and adding more features like a function for annotation reports, private annotations, votes, searching for annotations and more powerful filtering mechanism could make it a good system for use in further testing on users or as competitor in the web annotation arena. Some other features missing have been discussed in 7.1.1.

9.3 What happens to the new Web 2.0 systems?

As many new start up annotations systems are currently in the process of building a user base it will also be exiting to see if annotation sharing for web pages is ready for mass consumption. Just watching how and if services like Diigo, Trailfire, Stickis and Sharedcopy survive can tell us something about how many users it will take before a critical mass of users is reached and annotations are commonplace when visiting web pages. As Diigo and other systems covers the entire internet it is a very rare occurrence when someone has commented on a random page using a web annotations system at the moment. How many users does it take before it will be commonplace to find other peoples annotations on a page and actually make the systems usefully beyond as a personal annotation system?

9.4 Semantic annotations

With the developing of the semantic web and the wish to make computer understandable descriptions of resources, making such semantic descriptions work for annotations would give them much more interoperability. This would also enable search engines to be able to understand the annotations on a page and could make use of them.

The development of a semantic language for annotations is dependant on a standard for annotations emerging, either through the market of annotations solutions or a revival of work on Annotea.

9.5 Standardization

Like mentioned the annotation marked is at the moment flooded with solutions that don't work with each other. A standard seems unlikely to happen in the near future. A future project might be to evaluate what would be needed to bring more standardization or at least how different services can be interoperable through use of open API's. This has to some degree happened in the more mature social networks, but is still lacking for annotations.

Bibliography

[1] Marshall, C. C. 1997. Annotation: from paper books to the digital library. In *DL '97: Proceedings of the second ACM international conference on Digital libraries*, 131–140, New York, NY, USA. ACM. 1, 5, 11, 14, 15

[2] Marshall, C. C. & Brush, A. J. B. 2004. Exploring the relationship between personal and public annotations. In *JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries*, 349–357, New York, NY, USA. ACM. 1, 5, 14, 22, 44, 47

[3] Bush, V. 1945. Asas we may think. *Atlantic Monthly*. 1

[4] Esposito, J. J. 22.01.2008 2003. The processed book. *www.firstmonday.org*, 8(3). 2, 8, 13, 15, 16

[5] G. Lebow, D. & W. Lick, D. 2005. Hylighter: An effective interactive annotation innovation for distance education. In *20th Annual Conference On Distance Teaching and Learning*. 2, 6, 16, 18

[6] 2008. Diigo.com. Web site `http://www.diigo.com/about` Last accessed: 30/6/2008. 2

[7] Kahan, J. & Koivunen, M.-R. 2001. Annotea: an open rdf infrastructure for shared web annotations. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, 623–632, New York, NY, USA. ACM. 2, 5, 13

[8] Wilson, R. & Landoni, M. Eboni electronic textbook design guidelines. Technical report, University of Strathclyde, 2002. 2, 10

[9] Yee, K.-P. 2002. Critlink: Advanced hyperlinks enable public annotation on the web. In *ACM 2002 Conference on Computer Supported Cooperative Work*. 2

[10] Marshall, C. C. 1998. Toward an ecology of hypertext annotation. In *HYPERTEXT '98: Proceedings of the ninth ACM conference on Hypertext and hypermedia : links, objects, time and space—structure in hypermedia systems*, 40–49, New York, NY, USA. ACM. 5

[11] Wolfe, J. L. 2000. Effects of annotations on student readers and writers. In *DL '00: Proceedings of the fifth ACM conference on Digital libraries*, 19–26, New York, NY, USA. ACM. 5

[12] Nokelainen, P., Kurhila, J., Miettinen, M., Florien, P., & Tirri, H. 2003. Evaluating the role of a shared document-based annotation tool in learner-centered collaborative learning. In *Proceedings of The 6th IEEE International Conference on Advanced Learning Technologies*, 200–203. IEEE Computer Society Press. 6, 15, 47, 48, 50

[13] Wang, C.-Y. & Chen, G.-D. 2004. Extending e-books with annotation, online support and assessment mechanisms to increase efficiency of learning. In *ITiCSE '04: Proceedings of the 9th annual SIGCSE conference on Innovation and technology in computer science education*, 132–136, New York, NY, USA. ACM. 6, 15, 50

[14] Chisik, Y. & Kaplan, N. 2006. The social life of books in the humane library. In *JCDL '06: Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, 312–313, New York, NY, USA. ACM. 6

[15] Kaplan, N. & Chisik, Y. 2005. Reading alone together: creating sociable digital library books. In *IDC '05: Proceeding of the 2005 conference on Interaction design and children*, 88–94, New York, NY, USA. ACM. 6

[16] Glover, I., Xu, Z., & Hardaker, G. 2007. Online annotation - research and practices. *Comput. Educ.*, 49(4), 1308–1320. 6, 12, 13, 44, 47, 50

[17] Hwang, W.-Y., Wang, C.-Y., & Sharples, M. 2007. A study of multimedia annotation of web-based materials. *Comput. Educ.*, 48(4), 680–699. 7, 14, 15, 47, 50

[18] Brusilovsky, P. & Rizzo, R. 2002. Map-based horizontal navigation in educational hypertext. ACM Press. 513345 1-10. 8

[19] Cabanac, G., Chevalier, M., Chrisment, C., & Julien, C. May 2007. Collective annotation: Perspectives for information retrieval improvement. In *RIAO 2007: Proceedings of the 8th International Conference on Information Retrieval - Large-Scale Semantic Access to Content (Text, Image, Video and Sound)*, Pittsburgh, PA, USA. 8, 12, 22, 46

[20] Baca, M., Abrams, W., Reece, M., Clinkscales, D., Thomale, B., & Wolfinbarger, M. *VitalSource Bookshelf*. VitalSource, 2007. 8

[21] Ames, M. & Naaman, M. 2007. Why we tag: motivations for annotation in mobile and online media. 9

[22] Wu, X., Zhang, L., & Yu, Y. 2006. Exploring social annotations for the semantic web. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, 417–426, New York, NY, USA. ACM. 9

[23] Cadiz, J. J., Gupta, A., & Grudin, J. 2000. Using web annotations for asynchronous collaboration around documents. In *CSCW '00: Proceedings of the 2000 ACM conference on Computer supported cooperative work*, 309–318, New York, NY, USA. ACM. 9, 14, 15

[24] Diaz, P. March 2003. Usability of hypermedia educational e-books. *D-Lib Magazine*, 9(3). 10

[25] Zhang, J., Tan, K.-C., & Chai, K.-H. September 2003. Systematic innovation in service design through triz. *Triz-journal*, 9(1). 13

[26] Hilden, J. 2008. Web annotation 2.0 - designing successful social web annotation systems. 13, 15

[27] Brush, A. J. B., Bargeron, D., Gupta, A., & Cadiz, J. J. 2001. Robust annotation positioning in digital documents. 13, 18, 28

[28] Tate, K. & Domb, E. April 1997. 40 inventive principles with examples. Ttriz-journal `http://www.triz-journal.com/archives/1997/07/b/index.html` Accessed: 19/8/2008. 18

[29] Network Working Group. January 1997. Hypertext transfer protocol – http/1.1. Internet standard. Available at `www.ietf.org/rfc/rfc2068.txt`. 26

[30] Kappe, D. October 2006. Using bookmarklets to disruptive effect. Agile Ajax Blog `http://blogs.pathf.com/agileajax/2006/10/using_bookmarkl.html` Accessed: 19/8/2008. 27

[31] Ippolito, B. December 2005. Remote json - jsonp. from future import blog `http://bob.pythonmac.org/archives/2005/12/05/remote-json-jsonp/` Accessed: 19/8/2008. 27

[32] Donovan, B. 2002. Everything can be a link with mozilla's window.getselection() method and the w3c dom range api. `http://dev.lophty.com/ahoy/article.htm` Accsessed: 20/05/2008. 29, 51

[33] Schütze, C. D. M. P. R. H. 2008. *Introduction to Information Retrieval*. Cambridge University Press. 39

A Installation instructions

Requirements from the server running Dannotation:

- Web server(only Apache tested) running PHP 4.3.x or newer
- MySQL database

The source needed to run it can be obtained from `http://www.stud.hig.no/~010378/thesis/dannotationp.zip`. The password for the zip file is "5H9z5e8QfGmQ".

To install on the web server copy the files to a folder called dannotation.

To make the database work the SQL script in `sqlcreate.sql` must be run in a blank MySQL database. The database name, database user, database password, database host and the URL to the dannotation folder as accessible from the internet must then be configured in the `include\config.php` file.

If everything went correctly this should get your own Dannotation server up and running. Go to the the root folder of Dannotation with your web browser and the general usage instructions should appear.

The live version of version of Dannotation is running at: `http://www.stud.hig.no/~010378/thesis/dannotation`.

B Core source code

B.1 Libraries and other code used

All libraries and functions used are under some form of open source licence. The exact licences where details could be found are listed in the source code along with the functions. Only the bigger libraries are listed here, other utility functions are credited in the source code.

- jQuery 1.2.3 from `www.jquery.com`
- Json2.js from `http://www.JSON.org/js.html`
- Services_JSON from `http://pear.php.net/pepr/pepr-proposal-show.php?id=198`

B.2 The bookmarklet

This is the full code for the bookmarklet. It is all one line in the bookmark, so it has been pretty printed for display here to make it more readable.

Listing B.1: The bookmarklet

```
javascript:(
  function(){
    j=document.getElementById('dannotation-js');
    if(j){
      alert('Script already loaded');
      return;
      }
    else{
      s=document.createElement('script');
      s.type='text/javascript';
      s.src='http://www.stud.hig.no/~010378/thesis/dannotation/
          dannotation-js.php?nocache='+Math.random();
      s.id='dannotation-js'; document.body.appendChild(s);
    }
  }
)();
```

B.3 Obtaining the range

This code is run on each `mouseup` event to check if the users has selected anything. If something is selected a range is created from the selection and stored. The stored range is then used to retrieve the data sent to server when the user makes a decision if he wants to add a highlight or comment. The code for this retrieval of the parameters stored about the range can be found in part 5.3 or the full code in the `annoloader.addHiglight()` function in the source code.

Listing B.2: Code to extract the range from user selection and store it

```
var selection = window.getSelection() ;
if (selection.isCollapsed) {$(".dannotation_status").html("
    Selection␣is␣blank"); return false;} //if nothing is
    selected, we quit and inform user
var range;
if (selection.getRangeAt)
  range = selection.getRangeAt(0);
else { // Safari, because it does not suport getRangeAt. Ref:
    http://www.quirksmode.org/dom/range_intro.html#t00
  range = document.createRange();
  range.setStart(selection.anchorNode,selection.anchorOffset);
  range.setEnd(selection.focusNode,selection.focusOffset);
}
// Source for flip code: http://blog.versionate.com/2007/04/25/
    getting-range-objects-from-dom-selection/
if( range.collapsed != selection.isCollapsed ) { //If the user
    selects backwards the range does not support this direction
    ,. This flips the elements
  range.setStart( selection.extentNode, selection.extentOffset )
      ;
  range.setEnd( selection.baseNode, selection.baseOffset );
  $(".dannotation_status").html("Selection␣was␣backwards.␣It's␣
      now␣flipped");
}
if (range.startContainer.parentNode != range.endContainer.
    parentNode) {          //abort if selection spans several
     parents.
    $(".dannotation_status").html("Selection␣must␣be␣contatined␣
        within␣parent␣element");
    return false;
}
if (range.startContainer.parentNode.tagName == "dannotation" ||
    range.startContainer.parentNode.tagName == "DANNOTATION") {
    //Some browsers UPPERCASE. Others lowercase....
  $(".dannotation_status").html("Selection␣can't␣start␣or␣end␣in
      ␣another␣higlight");
  return false;
}

annoloader.currentRange = range;    //sets the range
selection.collapseToStart;          //removes the users
    highlight and sets cursor at the beginning of section
annoloader.loadPopup(e.pageX, e.pageY); //load a pop-up menu
    with "highlight" or "comment" at cursor position

return false;  //prevent event bubble by returning false. Else
    it triggers for all elements bound over the current one
```

B.4 Marking text

This is the function to highlight text and create <dannotation> tags in the HTML documents. It takes the recreated range sent back from the server and it's index in the array of annotations as input. This marking function runs once for each of the annotations retrieved from server. It is called from the callback function to add annotations and from the callback function to load annotations when the page loads or updates.

If it fails to place the annotation in the document it adds it to the array of orphans. At the moment nothing is done with the orphans in Dannotation except storing them and printing an error in the browsers error log(The `say()` utility function prints to Opera's error log, Firebug for Firefox if present, Firefox error log if not and gives an alert prompt in other browsers).

Listing B.3: The function to mark text

```
annoloader.markText = function(highlight, highlightIndex) {

  //marking is based on Ahoy http://dev.lophty.com/ahoy/article.
      htm
  if((highlight.startElementParent != null)&&(highlight.
      parentIndex != null)){
    try {
      var parents=document.getElementsByTagName(highlight.
          startElementParent);
      var theParent=parents[highlight.parentIndex];

      if(theParent.childNodes[highlight.startNodeIndex] &&
          theParent.childNodes[highlight.startNodeIndex])  {
        var startNode = theParent.childNodes[highlight.
            startNodeIndex];
        var endNode = theParent.childNodes[highlight.
            endNodeIndex];
      }

      //create the range from our elements
      var range = document.createRange();
      range.setStart(startNode,highlight.startOffset);
      range.setEnd(endNode,highlight.endOffset);

      var insertSpan = document.createElement("dannotation");

      if (highlight.username == annoloader.username)  {   //if
          this is our highlight
        insertSpan.setAttribute ("class", "
            dannotation_my_higlight");
        insertSpan.style.backgroundColor = "#edeef9";
        insertSpan.setAttribute ("id", highlight.annotationId);
        insertSpan.setAttribute ("username", "Me");
        insertSpan.setAttribute ("arrayIndex", highlightIndex);
      }
      else  {              //if it's someone else's public
          highlight
        insertSpan.setAttribute ("class", "
            dannotation_public_higlight");
        insertSpan.setAttribute ("id", highlight.annotationId);
        insertSpan.setAttribute ("arrayIndex", highlightIndex);
        insertSpan.setAttribute ("username", highlight.username)
           ;
        insertSpan.style.backgroundColor = "#edf4df";
      }

      range.surroundContents(insertSpan);   //sourrounds the
         range with a span
      unbindEvent();      //unbinds all events to prevent
         multiple bindings and functions running multiple times
      rebindEvent();      //to bind events to new span.
      $(".dannotation_status").html("Text␣higlighted");

      return 0;
    }
    catch(e) {
      say("Failed␣to␣create␣textrange␣because:␣"+ e);
      //creates orhaned annotations if range can't be created...
      if (annoloader.orphan) {
        annoloader.orphan.push(highlight);
        return 1;
      }
      else {
        annoloader.orphan = new Array();
        annoloader.orphan[0]= highlight;
        return 1;
      }
    }
  }
  else {$(".dannotation_status").html("Missing␣paramaters␣to␣
     find␣element"); return 0;}
  return 0;

}
```

B.5 The update client side code

The updating of annotations happens on each mouseup and before adding annotations to database. It finds the annotation with the highest ID on client and checks the server if it's the the highest ID there too. If it's not the server sends back the new annotations and the callback function uses the `annoloader.markText` function to mark the new annotation(s).

Listing B.4: The update function on client

```
annoloader.update = function() {

  if(!annoloader.JSONObject) {   //the page is blank when we
     update. Do same as first check for annotations
      var url = "<?php␣echo␣$server?>/dannotation/dannotation.
         php?";
      url += "loadAnno=1&url="+Url.encode(location.href);
      annoloader.addScript(url);
  }
  if (annoloader.JSONObject) {
    var numAnno=annoloader.JSONObject.length;
    annoloader.locked = 1; //lock client while updating
    $(".dannotation_client_status").text("Client␣locked␣while␣
        updating␣annotations...");
    var i=0;
    var lastId= 0;
    //loop through the annotations to find the highest id on
        client.
    for(i=0; i<numAnno; i++) {
      if (annoloader.JSONObject[i].annotationId*1 > lastId) //
          annotationId is a string after return from server. *1
          casts to number as we are doing number operation
        lastId=annoloader.JSONObject[i].annotationId*1; // *1
            casts to number
    }
    //addScript loads the annotations from database higher than
        lastId
    annoloader.addScript("<?php␣echo␣$server?>/dannotation/
        dannotation.php?update="+lastId+"&url="+Url.encode(
        location.href));
  }

  else {
    $(".dannotation_server_status").text("Didn't␣find␣any␣
        updated␣annotations␣for␣this␣page");
  }

}
```

C Database model

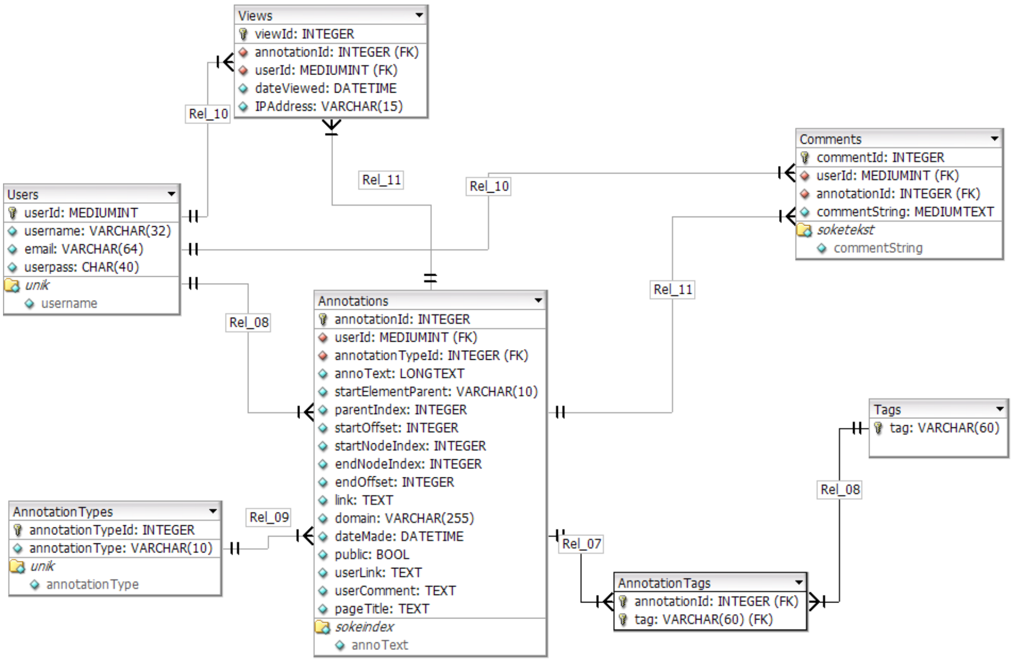
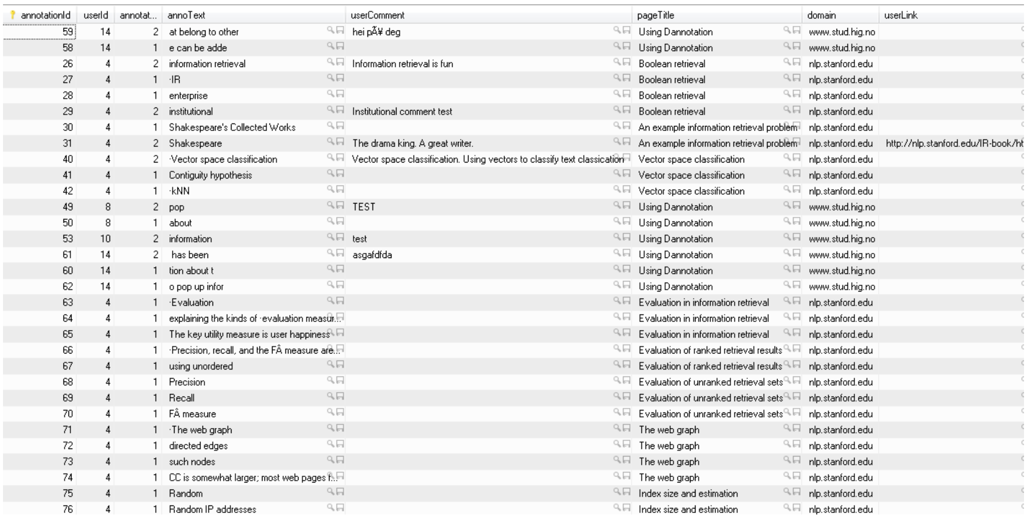


Figure 18: Model of the simple MySQL database

D Test data from the users



annotationId	userId	annotat...	annoText	userComment	pageTitle	domain	userLink
59	14	2	at belong to other	hei pÃ¥ deg	Using Dannotation	www.stud.hig.no	
58	14	1	e can be adde		Using Dannotation	www.stud.hig.no	
26	4	2	information retrieval	Information retrieval is fun	Boolean retrieval	nlp.stanford.edu	
27	4	1	·IR		Boolean retrieval	nlp.stanford.edu	
28	4	1	enterprise		Boolean retrieval	nlp.stanford.edu	
29	4	2	institutional	Institutional comment test	Boolean retrieval	nlp.stanford.edu	
30	4	1	Shakespeare's Collected Works		An example information retrieval problem	nlp.stanford.edu	
31	4	2	Shakespeare	The drama king. A great writer.	An example information retrieval problem	nlp.stanford.edu	http://nlp.stanford.edu/IR-book/ht
40	4	2	·Vector space classification	Vector space classification. Using vectors to classify text classication	Vector space classification	nlp.stanford.edu	
41	4	1	Contiguity hypothesis		Vector space classification	nlp.stanford.edu	
42	4	1	·kNN		Vector space classification	nlp.stanford.edu	
49	8	2	pop	TEST	Using Dannotation	www.stud.hig.no	
50	8	1	about		Using Dannotation	www.stud.hig.no	
53	10	2	information	test	Using Dannotation	www.stud.hig.no	
61	14	2	has been	asgafdfda	Using Dannotation	www.stud.hig.no	
60	14	1	tion about t		Using Dannotation	www.stud.hig.no	
62	14	1	o pop up infor		Using Dannotation	www.stud.hig.no	
63	4	1	·Evaluation		Evaluation in information retrieval	nlp.stanford.edu	
64	4	1	explaining the kinds of ·evaluation measur...		Evaluation in information retrieval	nlp.stanford.edu	
65	4	1	The key utility measure is user happiness		Evaluation in information retrieval	nlp.stanford.edu	
66	4	1	·Precision, recall, and the FÂ measure are...		Evaluation of ranked retrieval results	nlp.stanford.edu	
67	4	1	using unordered		Evaluation of ranked retrieval results	nlp.stanford.edu	
68	4	1	Precision		Evaluation of unranked retrieval sets	nlp.stanford.edu	
69	4	1	Recall		Evaluation of unranked retrieval sets	nlp.stanford.edu	
70	4	1	FÂ measure		Evaluation of unranked retrieval sets	nlp.stanford.edu	
71	4	1	·The web graph		The web graph	nlp.stanford.edu	
72	4	1	directed edges		The web graph	nlp.stanford.edu	
73	4	1	such nodes		The web graph	nlp.stanford.edu	
74	4	1	CC is somewhat larger; most web pages f...		The web graph	nlp.stanford.edu	
75	4	1	Random		Index size and estimation	nlp.stanford.edu	
76	4	1	Random IP addresses		Index size and estimation	nlp.stanford.edu	

Figure 19: The results from the test persons. "Annotat..." is the annotation type, 1 for highlight and 2 for comment

E Interview questions

Did you get any use out of the system?

What is useful about it?

How did you use it?

How hard was it to use the bookmarklet?

Could another solution have been better?

Any function you missed in the system?

What functions did you like?

What was the most useful function?

Useful to increase learning benefit?

Useful to increase cooperation?

Did it disturb reading?

How did you use it compared to marking on paper?

Did the fact that notes were shared dissuade you from make annotations?

How much of a barrier was logging into the system?

To the teacher students: something you would have used for your students?

F Email correspondence with the users

F.1 Email to the master students

Help with testing prototype for master thesis

Hello fellow master students.

I'm doing my master thesis on the use of an annotation system on learning material. Part of this project is to get some users to test my prototype on a relevant book. As you are using an HTML textbook for IMT4491 I would be very grateful if you can test my prototype system while you are reading the course material.

Currently the system allows for you highlight text and add comments, tags or links to related material. It would be very helpful for me if you could try to use these functions as you are reading the book. As all annotations are shared, it should also help fellow students understand the material or find links to related resources.

Simple instructions on how to use it and a link to the bookmarklet needed to use it can be found at `http://www.stud.hig.no/~010378/thesis/dannotation` . All it takes to use it on the textbook is to click the bookmark you create on each page you want to see other students annotations or make your own.

Regards Daniel Wenner 06HMMTA

F.2 Email to the other students

Forelesningsnotater fra forelesning om JavaScript og info om ønsket hjelp med testing av min masteroppgave.

Den korte forelesningen jeg hold om JavaScript i dag finner dere i vedlegget eller online her:
`http://www.stud.hig.no/~010378/javascript/JavaScript.htm` (eksport fra powerpoint, så fungerer ikke optimalt i annet enn IE). Skal også be Monica legge den ut på Fronter.

Jeg driver i øyeblikket med min masteroppgave. Den omhandler deling av annotasjoner(kommentarer) og markeringer i elektroniske dokumenter. I den forbindelse trenger jeg noen studenter til å bruke systemet og gi tilbakemeldinger på det. Siden dere bruker noen kilder på nett(W3schools, diverse tutorials, php.net osv.) i faget til Monica hjelper det meg masse om dere kan bruke min prototype på sider det er hensiktsmessig mens dere leser til eksamen eller jobber med prosjekt.

Systemet lar dere dele kommentarer på HTML sider med andre som bruker det. Dine kommentarer, linker og tags til den delen av dokumentet vil da kunne sees av andre, så det kan f.eks brukes til å legge til ekstra informasjon, stille spørsmål til andre i klassen om noe er uklart eller linke til mer utfyllende informasjon om noe.

For å bruke systemet må dere først gå til
`http://www.stud.hig.no/~010378/thesis/dannotation/` og så bokmerke linken i punkt 2 i Firefox, Opera eller Safari. Derfra får man opp systemet ved å klikke på det opprettede bokmerke på en side dere vil legge til en kommentar på. Etter man logger på får man mulighet til å se andres kommentarer på siden eller legge inn egne.

- For å opprette en kommentar markerer man ganske enkelt den teksten man vil kommentere og velger enten "highlight" for å bare markere den eller "comment" hvis man vil legge til kommentar, link eller tags.
- Andres kommentarer og markeringer vises med lys grønn og dine med lys blå. For å se innholdet av en kommentar ta musepekeren over den samtidig som man holder skift inne.
- For å svare på en kommentar velg "Add reply".
- Filtrering av kommentarer på brukere kan gjøres via "filter" menyen.
- Menyen kan skjules ved å bruke minimerings knappen på menyen.
- Det er mulig å linke til andre kommentarer på samme siden ved hjelp av sikteikonet som ligger bak link boksen.
- Det ligger også en engelsk forklaring på bruk av systemet på hjemmesiden

Etter dere er ferdig med prosjekt og eksamen vil jeg sende et kort spørreskjema om bruken av systemet. Håper dere også kan ta dere noen minutters tid til å svare på det. Alle svar på det og kommentarer som lages med systemet vil bli anonymisert i rapporten og vil bli slettet etter rapporten er ferdig.

I øyeblikket er det ingen funksjon for å sende passord om dere skulle glemme det. Om det hender, så send meg en mail, så skal jeg generere et nytt passord. Ellers kan også andre spørsmål eller kommentarer sendes til meg på mail. Håper dere kan ha nytte av prototypen min i innspurten og at dere tar den i bruk. Lykke til med prosjekt og eksamen :)

Mvh Daniel Wenner