

CHALMERS



Web 2.0 School Software

Designing for Usability

Fall | 09

Joel Sandlund

The Author grants to Chalmers University of Technology and University of Gothenburg the non-exclusive right to publish the Work electronically and in a non-commercial purpose make it accessible on the Internet.

The Author warrants that he/she is the author to the Work, and warrants that the Work does not contain text, pictures or other material that violates copyright law.

The Author shall, when transferring the rights of the Work to a third party (for example a publisher or a company), acknowledge the third party about this agreement. If the Author has signed a copyright agreement with a third party regarding the Work, the Author warrants hereby that he/she has obtained any necessary permission from this third party to let Chalmers University of Technology and University of Gothenburg store the Work electronically and make it accessible on the Internet.

Web 2.0 School Software

Designing for Usability

Joel Sandlund

© Joel Sandlund, October, 2009

Examiner: Olof Torgersson

Department of Computer Science and Engineering

Chalmers University of Technology

SE-412 96 Göteborg

Sweden

Telephone + 46 (0)31-772 1000

Göteborg, Sweden October 2009

ABSTRACT

A school in Gothenburg has an old program for handling their data about students. They have requested new software to be developed in order to support their need of archiving, structuring and accessing information about e.g. class attendance, *individuell utvecklingsplan (UIP)*, journals, grades etc.

The new system should be available through the internet with a web interface i.e. no installation should be necessary in order to access the system from a new computer.

The goal in this thesis is to design and implement the web 2.0 application that the school requested, in a MVC framework called Struts2, with emphasis on the interaction design of the system.

In order to design for usability a number of design methods were implemented. Also two usability tests were conducted with system mock-ups, to evaluate the users' requirements.

<u>ABSTRACT</u>	<u>3</u>
<u>ABBREVIATIONS AND TERMINOLOGY</u>	<u>7</u>
<u>1 INTRODUCTION.....</u>	<u>8</u>
1.1 BACKGROUND	8
1.2 THE PROBLEM	9
1.3 DOCTUM.....	9
1.3.1 SYSTEM DESCRIPTION.....	9
1.3.1.1 The Model	10
1.3.1.2 System Task Description.....	10
1.3.1.3 User Groups	11
1.3.2 TECHNICAL DESCRIPTION	11
1.3.2.1 Struts2	11
1.3.2.2 YUI.....	12
1.4 WEB 2.0 AND AJAX.....	12
1.5 LIMITATIONS	13
1.6 VERIFIABLE GOALS.....	13
<u>2 THEORY.....</u>	<u>14</u>
2.1 WHAT IS USABILITY?.....	14
2.2 AUTOMATIC SAVE	14
2.3 USABILITY TESTING.....	15
<u>3 METHOD</u>	<u>17</u>
3.1 DESIGN STRATEGY.....	17
3.2 STATE GOALS.....	18
3.3 CHARACTER PROFILES	19
3.4 PAPER PROTOTYPE AND STORYBOARD	19
3.5 USER STUDY.....	20
3.5.1 PARTICIPANTS	20
3.5.1.1 Who should participate?	20
3.5.1.2 How many should participate?.....	20
3.5.1 FIRST USER TEST	20
3.5.1.1 Goal.....	21
3.5.1.2 Context.....	21
3.5.1.3 Observation and Question based.....	21
3.5.1.4 Think Aloud	21
3.5.1.5 Interview	21
3.5.2 SECOND USER TEST.....	22
3.5.2.1 Goal.....	22
3.5.2.2 Context.....	22
3.5.2.3 Question based study	22
3.5.2.4 Questionnaire.....	22
<u>4 REALISATION</u>	<u>23</u>

4.1 CHARACTER PROFILES	23
4.2 DESIGN DECISIONS	23
4.2.1 HIERARCHICAL STRUCTURE.....	23
4.2.2 MENU	24
4.2.3 BREADCRUMBS.....	25
4.2.4 CALENDAR.....	25
4.2.5 SAVE BUTTON.....	26
4.2.6 FEEDBACK	26
4.2.7 COLOURS.....	27
4.2.8 MULTIPLE VIEWS.....	27
4.3 PAPER PROTOTYPE AND STORYBOARD	28
4.4 USER STUDY.....	28
4.4.1 FIRST USER TEST	28
4.4.1.1 System Description.....	28
4.4.1.2 Test Description.....	28
4.4.1.3 Evaluation.....	29
4.4.2 SECOND USER TEST.....	29
4.4.2.1 System Description.....	29
4.4.2.2 Test Description.....	30
4.4.2.3 Evaluation.....	30
<u>5 RESULT</u>	<u>31</u>
5.1 CHARACTER PROFILES	31
5.2 DESIGN DECISIONS	31
5.2.1 HIERARCHICAL STRUCTURE.....	31
5.2.2 MENU	32
5.2.3 BREADCRUMBS.....	33
5.2.4 SAVE BUTTON.....	34
5.2.5 FEEDBACK SYSTEM.....	34
5.3 PAPER PROTOTYPE AND STORYBOARD	35
5.4 USER STUDY.....	36
5.4.1 FIRST USER TEST	36
5.4.1.1 System Architecture.....	36
5.4.1.2 Reminder View.....	36
5.4.1.3 Orientation and Navigation.....	37
5.4.1.4 Attendance View.....	37
5.4.2 SECOND USER TEST.....	37
5.4.2.1 Slider Button.....	37
5.4.2.2 Comment Field	38
5.4.2.3 Feedback system	38
5.4.2.4 Save Button or Auto save	38
5.4.2.5 Quick Buttons.....	38
5.5 FINAL SYSTEM DESCRIPTION.....	39
<u>6 DISCUSSION</u>	<u>41</u>
6.1 USER STUDY.....	41
6.2 VERIFIABLE GOALS	42
6.3 CONCLUSION.....	43

REFERENCES	44
INTERNET REFERENCES.....	44
APPENDIX A	46
USER TEST 1	46
OBSERVATIONS/QUESTION ANSWERS	46
USER TEST 2	48
QUESTIONNAIRE.....	48
ANSWERS.....	49
APPENDIX B.	51
SYSTEM SCREENSHOTS OF THE SECOND USER TEST.....	51

ABBREVIATIONS AND TERMINOLOGY

MVC	Model View Controller
HCI	Human Computer Interaction
GUI	Graphical User Interface
IUP	Individuell Utvecklingsplan (Individual development plan)
SAA	Stand alone application, an application that is installed in an operating system to perform a specific task.
Branch tree	Tree structure used to represent the architecture of e.g. a web page.
Cross-links	A link from one branch of a tree to another.
Pagination	Way of browsing sets of data in tables.
Bread crumbs	An indicator of where -in a web page -the user is currently at.

1 INTRODUCTION

During the past few years Swedish schools has gotten more demands on their shoulders to archive student information such as grades, individual judgements, class attendance etc. This kind of information has historically been stored on sheets of paper, spreadsheets and databases.

At the same time web 2.0 has introduced some major changes in the way web applications are used, thought of and developed. Much of the traditional office software has migrated to the web, increasing their availability and lowering maintenance cost.

1.1 Background

Freinetskolan Bild&Form is a private school in Gothenburg Sweden with around 200 students in the age ranging from 7 to 16. It sticks out from other schools by focusing on art and creativity. All its students are taking one or more courses in art, design, music etc.

As a school with a relatively high amount of creative works, it has to have a sturdy structure for archiving and organizing material as such. Thus the school has been -since it was founded in 1998 -a big friend of IT, which has provided them with that structure.

Ever since the school started using IT as an *archiving tool* (their first file server was installed in 1999) the computers have constantly gained more and more land. Before that data was put in to separate spreadsheets and paper folders (see Figure 1). Today every class has computers on which the students can log in to their server accounts and browse their portfolio.

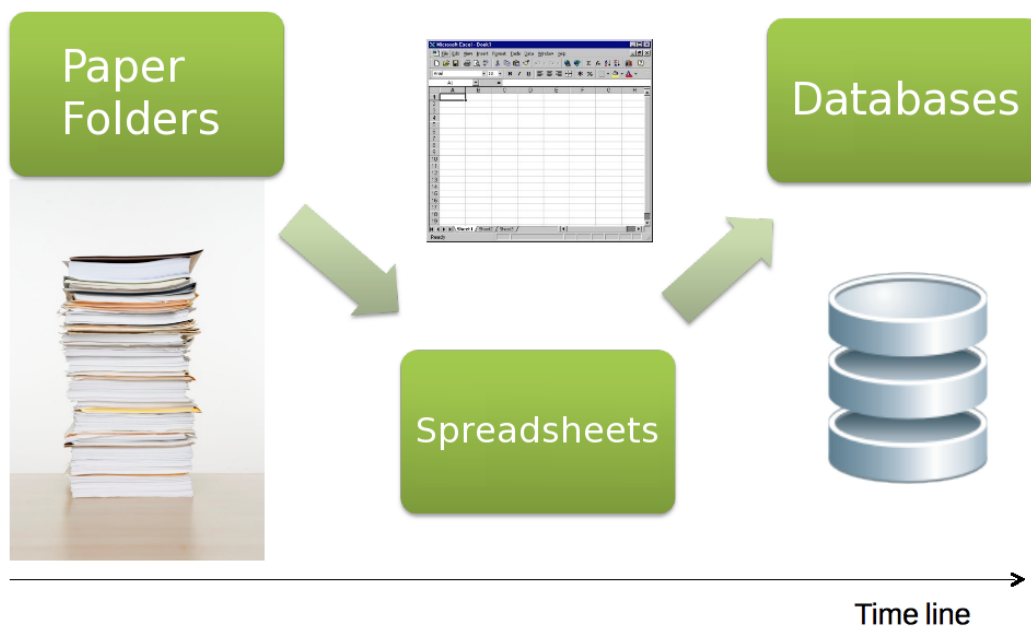


Figure 1, Information Handling History

1.2 The Problem

Today the school has two separate information systems:

1. *B.* is the system that handles information about the students' grades, class attendance, journals and individuell utvecklingsplan (IUP). It has to be accessed through a main frame computer that hosts the database locally, why the old system has a low rate of availability and teachers first have to write down the class attendance on paper and later in the day sit down by the main frame and log it into the database.
2. The school web page has replaced much of the broadcasting communication between teachers and parents, which previously was done with weekly mails or papers that the students carried home to their parents. Yet, the communication is only one way (*from* teachers *to* parents) and the file server information with portfolios, grades, attendance etc. is still separated from the Internet.

The personnel want *B.* to have a *higher rate of availability* so that they can access the system from their laptop during a class or from home when the school day is over. In order to achieve this there are basically two options: 1) the accessing system has a *stand alone application (SAA)* client with a graphical user interface (GUI) installed, or 2) the system has a web interface with which the user can perform his tasks.

The second request from the school is that the teachers should get better access to content in the system such as their student's rate of attendance, grades, IUP and more. If the school should be able to rely on that the parents can view the content they couldn't have to have a SAA client installed too. The web interface should then be used with the advantage of higher availability among parents.

A proposed system called Doctum was designed based on these premises. Doctum's main goal is to do the tasks that *B.* already did, but with a web interface and with a more user centred design. This would in reality mean that system 1 and 2 are merged together to one system.

1.3 Doctum

The system that is going to carry the school in to the web 2.0 era, is the application that this thesis aims to develop a part of. *Doctum* is a web application built in the *web application* framework *Struts2* (see section 1.3.2.1 *Struts2*).

Doctum should be accessible from the teacher's desktop computer during class, which poses some immediate demands on the user interface -from a usability perspective -as well as on the systems availability and security.

In a longer perspective the system should be able to handle student grades, individual judgement, IUPs etc. Thus the system has to be extendable to fit in additional modules as time goes on.

1.3.1 System Description

This section will describe what the system does, what model it uses and what technical frameworks and tools are being used.

1.3.1.1 The Model

Due to the high demand on the system's availability, Doctum was decided to run on a dedicated server that could be accessed through the cloud by computers and handheld devices. Naturally in this case a *client server model* will be used (see Figure 2).

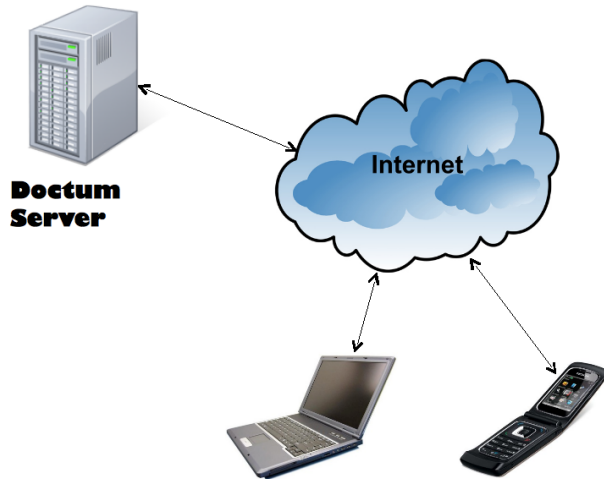


Figure 2, Client Server Model

1.3.1.2 System Task Description

Generally all web applications with multiple user accounts need the following components:

- A database in which the users and *request objects* (data which the users can access with permission) are stored
- A list of the users which access the system
- A user access permission system for objects and directories that keeps track of and enforces the users rights to sets of information stored in the database
- A session controller that keeps track of which users that are logged in and gives them access to their request objects

These components are included in Doctum's base system. But so far the system is just a web platform like any other in the bunch. The goal with Doctum's is to have a full coverage of the school's IT information handling requirements, thus it was designed to be extendable with plugins such as:

- Attendance
- Grades
- Journals
- IUPs
- Schedule
- Course information
- Weekly letters
- Food menu
- Application queue

Since it's neither wise nor feasible for a project of this scale to include all these functions at once, this thesis is going to focus on the first-mentioned item, namely attendance.

The parts of the system, which will be implemented in this thesis, are seen in Figure 3.

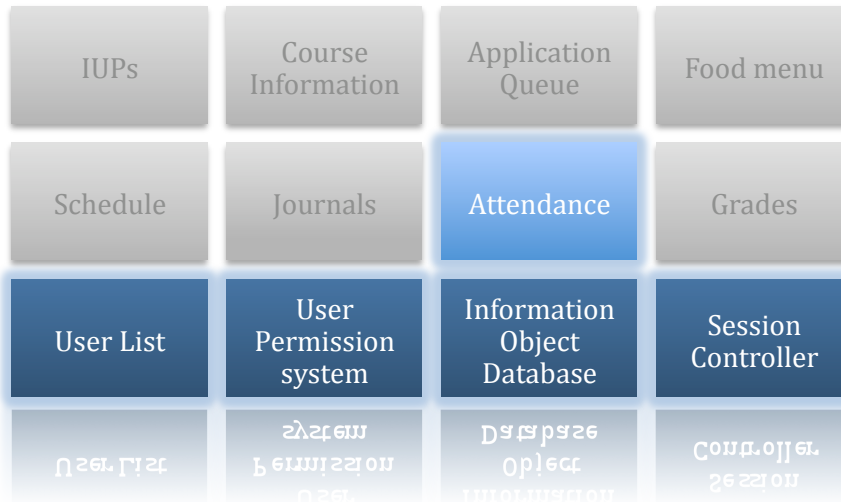


Figure 3, System Modules

1.3.1.3 User Groups

The primary user group is the *school personnel*, teachers in particular but also office employees; they are the users that provide the system with content whether it is information about a student’s attendance or a weekly letter to the parents. But they are not to be mistaken for exclusive *content providers* because they are also recipients of the published information e.g. a teacher might get information about a schedule update from the office.

The secondary user group is the parents/students. The group is considered secondary because it will be supported by the system in a later stage and it is almost exclusively a recipient of information which makes it practically a subgroup of the school personnel group regarding the user interface because the functions that they have to have access to is a subset of what the personnel is using.

1.3.2 Technical Description

This section will give a brief description of the most important technical aspects of the system.

1.3.2.1 Struts2

Struts2 is a MVC (Model View Controller) web framework (sometimes also called a Model2 framework. It is a merge between *Apache Struts* (released in 2001) in which one of the primary goals was to incorporate the MVC pattern from desktop applications to web software, and *WebWork*, an OpenSymphony project that itself was a fork from Apache Struts code base.

Some of Struts2’s main features are:

- *Java*: The controller component in Struts2 is written in Java. Being a mature programming language with a large class library this is a huge bonus when building complex controllers.

- *Plug-ins*: Rather than having the core framework include everything, Struts2 can plug in third-party libraries, which makes it easier for the developer to tailor the libraries used for the application.
- *Testability*: Testing actions and classes in Struts2 is very easy which helps debugging when the project has become large and complex.
- Struts2 MVC structure provides a powerful way to adapt the *view layer* according to what device that requests the information; hence it could be extended to offer tailored views for handheld devices.

1.3.2.2 YUI

One of the more important Struts2 plug-ins that was used for Doctum is YUI (Yahoo User Interface, <http://developer.yahoo.com/yui>). It provides a big library of Javascript *widgets* ranging from simple push buttons to fully-fledged *rich text editors* and *slider buttons*.

YUI also has a utility called *Connection Manager* that has a simplified interface to the AJAX XMLHttpRequest object, which enables asynchronous updates of pages and is frequently used throughout the whole application.

1.4 Web 2.0 and AJAX

Before going into a detailed level of Doctum, one question needs to be answered, namely “what is Web 2.0?” As it turns out, this is a quite difficult question to answer.

Web 2.0 is a term introduced in 2004 to characterize design patterns in a constellation of new generation Web applications which may provide an infrastructure for more dynamic user participation, social interaction and collaboration. (CSA)

One of the technical characteristics that is intimately associated with web 2.0 is *AJAX*. Except for the advances of *dynamic* scripting languages that have meant a whole lot for the new web, a Web 2.0 application wouldn’t have been possible to accomplish without the use of *AJAX* (Asynchronous Javascript and XML). In fact, from a programming perspective Web 2.0 is synonymous with *AJAX*.

The term *AJAX* was coined in February 2005 by Jesse James Garrett and is used to describe the interaction between many technologies. At the core is the XMLHttpRequest object, which is supplied by the web browser. (Roughley I)

AJAX technologies provide a big step in usability for the web as they -if successfully used -can create an “illusion” of that a page is working as an application that is run on the client’s computer.

Rather than requiring the whole HTML page to be reloaded to make a server dependent change in an element, an asynchronous call (using the XMLHttpRequest object) can be made to the server when the user performs an action that requires an object to be updated. When the server receives the request it will translate it and send back only the component of the page that was requested, making it visible to the user instantly which is time saving and increases workflow. The effect makes the seams between the server and client more transparent which makes the user experience less awkward.

1.5 Limitations

The work in this thesis will be limited to the *class attendance module* and the necessary infrastructure to make this possible. However, the system is constructed to be highly extendible with new modules, hence the design decisions are founded on the possibility of supporting a larger set of modules later on.

1.6 Verifiable Goals

- How can a web application for logging students' attendance be designed to support usability?
- Can *auto save* be used instead of a save button to enhance usability?

2 THEORY

The literature study aimed to give a picture of the research and enlightened opinion in interaction design, which has a relevant value to this thesis, and to find an optimal way to test the usability of the application in an efficient manner. In this chapter the literature that was found in the literature study is summarized and described.

2.1 What is usability?

In order to be able to discuss *usability* as something verifiable it is of substantial importance to determine what the word actually means; the concept of it. This is the ISO definition of the term usability:

- Usability: Extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.
- Effectiveness: Accuracy and completeness with which users achieve specified goals.
- Efficiency: Resources expended in relation to the accuracy and completeness with which users achieve goals.
- Satisfaction: Freedom from discomfort, and positive attitudes towards the use of the product. (ISO9241-11 3 Definitions)

Effectiveness, efficiency and satisfaction are the cornerstones of the usability concept and are all equally important, however generally designers have to make tradeoffs with these because of limited resources or time constraints.

2.2 Automatic save

One of the goals for this thesis is to investigate if *auto save* can be used instead of a save button to enhance usability?" Unfortunately the information, which can be read about the subject, is very sparse but there are some anticipations about it.

Applications should automatically save documents. For starters, when a user is done with a document and requests the Close function, the application should go ahead and write the changes out to disk without stopping to ask for confirmation with the Save Changes dialog box. (Cooper A)

Most applications (and web applications in particular) still use a save changes dialog box or a save button after a form. But there are some forerunners such as Apple (see Figure 4) and Google who have adapted automatic save already.

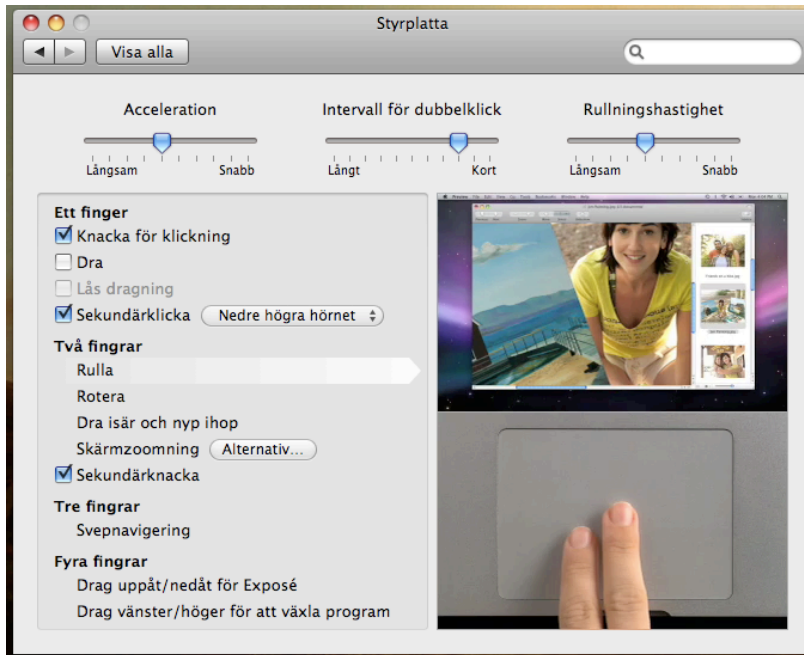


Figure 4, Mac OS uses automatic save for system configurations

It's important that this automatic save function be performed in such a way as to not affect the responsiveness of the user interface. Saving should either be a background function, or should be performed when the user has stopped interacting with the application. (Cooper A)

Following the same line of argument Cooper argues that unnecessary tasks should be minimized as much as possible, thus if the save button isn't needed and it doesn't have a technical function the user shouldn't be bothered with it, relieving him from unnecessary excise.

Excise is the extra work that satisfies either the needs of our tools or those of outside agents as we try to achieve our objectives. The distinction is sometimes hard to see because we get so used to the excise is being part of our tasks. (Cooper A)

Also when the data is automatically saved there would be less uncertainties "did I save the data before I logged out?" plus it would minimize the data loss if the client crashed.

Another aspect of auto save is that it takes away a privilege from the user. This can be both good and bad depending on how it's seen

One of the oldest guidelines for usable interaction design is to increase the user's sense of control and freedom. It feels good to be in control. It feels bad to be dominated by a machine" (Nielsen J)

The user loses some of his control but also loses unnecessary excise.

2.3 Usability testing

To evaluate the application that is designed and implemented in this thesis its usability has to be tested with real users. If properly conducted, a usability test can effectively determine aspects of usability such as:

- Naming: Do section/button labels make sense? Do certain words resonate better than others do?
 - Organization: Is information grouped into meaningful categories? Are items located in the places people might look for them?
 - First-time use and discoverability: Are common items easy for new users to find? Are instructions clear? Are instructions necessary?
- Effectiveness: Can customers efficiently complete specific tasks? Are they making missteps? Where? How often?
(Cooper A)

Before executing a user test there are a few important decisions to take into consideration, some of those are, what participants to use, what system artefact or mock-up to use, whether to conduct a qualitative or quantitative test and what test methods to use.

In About Face Cooper suggests that for most forward oriented usability tests a qualitative study is preferable before quantitative.

Data gathered by the hard sciences like physics are simply different from that gathered on human activities: Electrons don't have moods that vary from minute to minute, and the tight controls physicists place on their experiments to isolate observed behaviours are impossible in the social sciences. Any attempt to reduce human behaviour to statistics is likely to overlook important nuances, which can make an enormous difference to the design of products. Quantitative research can only answer questions about "how much" or "how many" along a few reductive axes. Qualitative research can tell you about what, how, and why in rich detail that is reflective of the actual complexities of real human situations.
(Cooper A)

Jacob Nielsen writes in his article "Why you only need to test with 5 users" that "The best results come from testing no more than 5 users and running as many small tests as you can afford". The correlation between the number of usability problems found and the number of participants in the study can be illustrated with the graph in Figure 5 (Nielsen J) according to Nielsen.

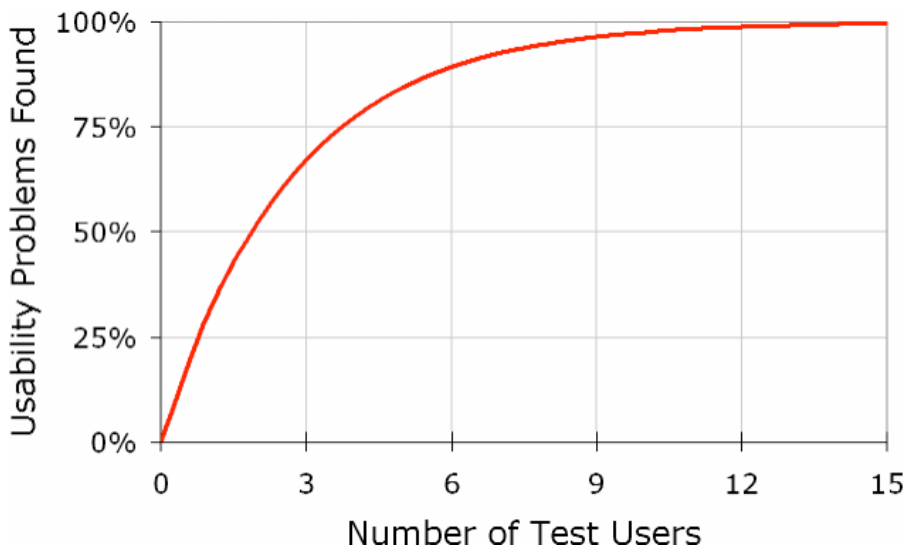


Figure 5, Participants and usability problems found

Nielsen explains further in the article "As you add more and more users, you learn less and less because you will keep seeing the same things again and again. [...] After the fifth user, you are wasting your time by observing the same findings repeatedly but not learning much new.

3 METHOD

This chapter will describe the methods that were used in the project and motivate why they were used. It begins with a section about design strategy, which places the methods in to a larger context of design phases.

3.1 Design Strategy

For selecting the methods much inspiration was taken from (“design methods, second edition”, John Chris Jones) in which many design methods are described and categorized in stages. The project can be divided into three stages: *divergence*, *transformation* and *convergence*, see Figure 6.

The names (divergence, transformation and convergence) are meant to refer more to the new problems of system designing than to the traditional procedures of architecture and of engineering design. Confusing and unhelpful as it may be to a professional designer to think of these three things as separated, there is little doubt that their separation is prerequisite to whatever changes of methodology are necessary at each stage before they can be reintegrated to form a process that works well at the systems level. (Chris Jones J)

The design process starts off with the divergence phase, which is characterized by that the problem boundary is undefined. The divergence phase usually contains methods for structuring and stating uncertain variables such as the objectives, expected user and limitations. Also it is appropriate to do a field study of what is done already in the area.

This term (Divergence) refers to the act of extending the boundary of a design situation so as to have a large enough, and fruitful enough, search space in which to seek a solution. (Chris Jones J)

Divergence soon passes into transformation, the stage in which generally the elaboration of ideas takes place. The abstract ideas takes physical form with help from mock-ups, paper prototypes and test implementations. The prototypes are tested on real users, and then become subject for redesign and the cycle continues throughout the whole transformation phase.

Ultimately, when the design has been determined, the project is wrapped up in the *convergence phase*, which is often characterized by an intense implementation period.

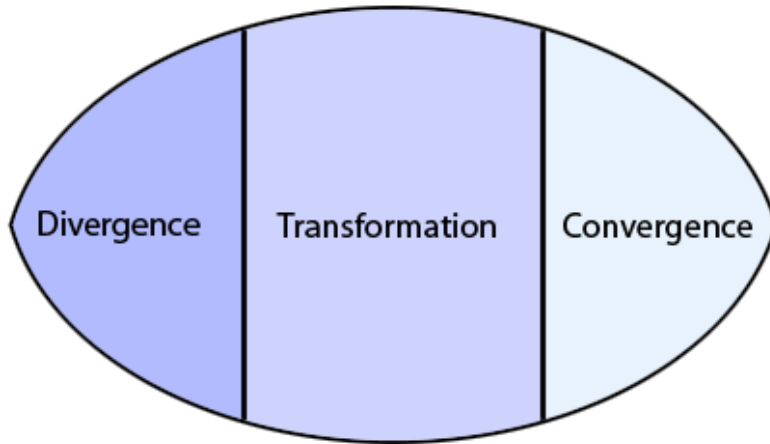


Figure 6, Three Stage Design Phase

The methods that follows in this chapter were implemented using these three categories, here is the order and classification in which they were practiced.

- Divergence
 - State Goals
 - Literature study
 - Character Profiles
- Transformation
 - Design choices
 - Paper Prototype and Storyboard
 - User Study
- Convergence
 - Implementation

Only the methods that are somewhat intricate or altered in a way among those listed above will be explained in further depth. Thus the methods that are straight forward enough or self-explanatory aren't described in further detail in this chapter.

3.2 State Goals

The primary goals was already established before the project was started but since this is a good way to initiate a project with the job requestor it was decided that a meeting for elaborating objectives should be scheduled before the design process started.

Another positive output from this method is that the goals become more nuanced and small misunderstanding may be cleared up before there is any harm done.

Last but not least, when the goals are concluded and written down, the paper can serve as a contract for everyone in the project so if there is a dilemma in values or if the work strays from the goals that were agreed upon it can be a quick reminder.

3.3 Character Profiles

In order to design for usability one has to know who the product is for. A detailed user analysis will not be implemented because the users' computer knowledge is very wide spread; there are young teachers which use computers in their work and everyday living as well as teachers who have been employed for twenty years that don't know the first thing about computers.

For the elaboration of the user profiles the guide "Ten Step to Personas" (2005) will be used. Nielsen suggests that one should gather knowledge about the different users through several different sources "interviews, observations, second hand information, questionnaires, reports, cultural probes etc." all these methods however will not be used but the first user test will serve as a way of implementing a few.

Before the character profiles can be created the *boundaries* and *situation* for the expected user has to be determined. Boundaries meaning: the users technical experience and domain knowledge. Situation includes expected interest, time to learn and use frequency.

In order to get a more creative design process and to be able to communicate user groups' requirements, this method should be implemented in the very beginning of the project. The main advantage that this method outputs is a more sharply defined profile of the typical user.

The early profiles will be used mostly during the initial design phase before the first user test, thereafter the profiles will be changed/updated with the input that is gathered from the real users.

The aim is to elaborate two or three character profiles before the GUI and functionality design is started. Utilizing profiles correctly it can speed up the design process as well as make decisions regarding usability more accurate.

The different profiles' abilities and disabilities should complement each other in a way so that all the expected user groups are accounted for.

3.4 Paper Prototype and Storyboard

A paper prototype is most often used for usability testing "representative users performs realistic tasks by interacting with a paper version of the interface that is manipulated by a person playing computer" (Paper prototyping, Carolyn Snyder) but it can also be used only by the designers to get a modifiable visual of the system which could be elaborated and also suggest the structure for the graphical interface. A paper prototype of a web interface is often called a wireframe.

In this project a paper prototype should be created along with a *storyboard*, which together is a powerful tool for the designers to get a broader view of the systems architecture and workflow.

A storyboard is a series of drawings or images that represents how and interface would be used to accomplish a particular task. It's basically a flowchart (...) they are typically used to understand the flow of the user's work and how the interface will support each step.
(Snyder C)

The paper prototype and storyboard are intended to be used within the development and not reviewed by users, but in the case of that the system wouldn't be ready for the first user test, the paper prototype could be used in its place.

3.5 User Study

Two user tests should be performed during the project to evaluate different aspects of the system architecture and how certain aspects of the GUI is perceived by the users, such as the feedback system and the save procedure.

3.5.1 Participants

When choosing participants there are two main questions to consider; “who and how many should participate in the survey?” These questions can be answered in a number of ways depending on the kind of study.

3.5.1.1 Who should participate?

The standard end user that the system is designed for is an employed teacher with generally high *domain knowledge* (knowing about what they need from the system) but with very uncertain technical experience.

It was decided that the users elected for the study should rather have high domain knowledge than technical experience in order to match the expected end user as well as minimizing the users learning period on the study occasions and to get more structured data.

Consumers with expertise will have more product-related information available in their memory. [...] When product-related information is available in memory it becomes possible to discriminate between relevant and irrelevant product information.
(Jan P. L. Schoormans)

Lead users suggested in (Hippel) would be excellent interviewees. Unfortunately they are hard to find and considering the time limitations they are beyond the reach for this project. Instead it was decided that focus should be directed on finding users with domain knowledge ranging from low to high technical experience.

The teachers working at the school (Freinetskolan Bild&Form) have achieved a high level of domain knowledge by long experience from logging their students' attendance with different methods, thus it provide a great resource for a *representative sampling* based on technical skill. The group of teachers is also very variably regarding technical experience; therefore the elicited users can be selected on the basis of their technical skill.

3.5.1.2 How many should participate?

There are many different opinions about how many interviewees are sufficient for a user study such as this one. However, much of the literature that came up in the literature study (covered in the theory chapter) was clear-cut towards “lesser participants, more tests”.

Thus it was decided to perform a *qualitative study* (as in opposite to *quantitative*) with fewer test persons but with deep probing interviews and longer test periods. A group of six people should be elicited based on how they rated themselves on a scale with regard to their technical experience. All the test subjects should be working as teachers with kids in the same age (13-16) to form a *representative* user group.

3.5.1 First User Test

This user test takes place during the early stages of implementation. It was decided to use the actual system as mediating tool because this would give the best user experience in terms of responsiveness and look and feel. In worst case (if the implementation would

have lagged behind to the extent that it couldn't serve as mediating tool) the paper mock-up could be used in the systems place.

Neither interface nor functionality for setting attendance on a student will be near completion in this stage of development. Thus system architecture and naming of hyper links will be the main focus for the test.

3.5.1.1 Goal

The goal with this user test is to evaluate if the thought *system architecture* and the way of navigating it is satisfying for the user. The questions will be focused on the test subjects' mental model of the system such as the system architecture/model, links/link names, the view for setting attendance and the reminder view.

If an inconsistency in the system can be discovered in this early stage of the design it will be much less time consuming to do the necessary changes than later on.

Even though it is primarily a test of the system, it is also a test of the users meaning the test will prove if some of the assumptions made in the user analysis method and character profiles method were legitimate. If not they have to be altered in order to match the test users' profiles.

3.5.1.2 Context

The test subjects are invited to a quiet room in the school facilities. They get a 10 minutes PowerPoint presentation of what the system is for and how it will differ from the system they are currently working with. The PowerPoint doesn't reveal any screenshots or graphics of the actual system.

Afterwards the users get to log in to the system while a *think aloud test* is performed for 15-20 minutes. When this test has finished it is followed up with a short interview. All in all, the test is expected to last no more than 40 minutes.

3.5.1.3 Observation and Question based

This test is conducted in one observation based and one question based phase. These methods are often talked about as two different principles but they also complement one another effectively. Optimally two tests should be conducted with one of each method but instead of choosing one or the other it was decided to split the test in two phases to combine both strengths.

3.5.1.4 Think Aloud

The think aloud test is performed by a test subject and a person conducting the test. The user will be given a number of tasks like "log in and navigate to the page for setting student attendance". While the test subject is performing the task he will be asked to talk about how he perceives and interpret every aspect of the system. The test conductor should make a note of everything the user say ("I cant find the button") and do ("the user looks like he's searching for a button").

Meanwhile the conductor should make as little noise as possible in order not to affect the user in a way that could be negative to the test. Sometimes though, it could be appropriate to remember the user of the task or ask him to clarify a thought.

3.5.1.5 Interview

The interview is conducted immediately after the think aloud test because the user will soon forget smaller details and impressions. The interview should follow an interview

guide that complements the tasks that the user was asked to perform in the think aloud test.

3.5.2 Second User Test

This test will be conducted in the later stages of development and focus primarily on the GUI for setting attendance on students. The real system will be used as mediating tool with the same setup as in the first test but with additional GUI and functionality for registering and viewing attendance.

3.5.2.1 Goal

This test is made to evaluate different usability aspects of the view for setting student attendance. There are a number parts of this system that have to be tested.

- The slider button for setting the amount of time the student has been absent from the class
- The text field in which the teacher can write comments about a student
- The pagination between pages
- The feedback system which notifies the user of the save status of his changes
- The auto save mechanism

3.5.2.2 Context

The first test lasted for roughly 40 minutes whereas this test will have a span over fourteen days. The test period is introduced with a presentation where the whole test group participates. On this occasion the whole system will be demoed and they will get to ask questions.

The teachers will get their own account names and passwords with which they can log in to the system through the web page.

As an incentive for the teachers to use Doctum during the test period authentic data will be applied to their account, hence the lectures on which they should register attendance is actually conformable with their real schedule and their own students.

The test subjects are encouraged to use the system in different environments, especially during class.

3.5.2.3 Question based study

In opposite to the first test in which the whole system architecture should be evaluated, this test is conducted to answer a few specific questions, which focuses on the GUI of the attendance page.

The questions will be collected in a questionnaire, which will be handed out to the participating test subjects.

3.5.2.4 Questionnaire

The questionnaire will contain around five questions, which focus on the main interaction subjects in the attendance view; the feedback system, save button, pagination, slider button and the comment field.

Open-ended questions were chosen for the most part because the user group is relatively small hence a statistical survey is not feasible. Also the answers from open-ended are more exhaustive than closed-ended questions.

4 REALISATION

In this chapter the actual work process is described and analyzed along with the output of the methods that were used. The most important design decisions are discussed in 4.2. The chapter ends with a discussion about how the time plan was followed and how different methods worked out.

After the methods were chosen and sketched out they had to be implemented. The described methods were implemented in the divergence and transformation phase. Some methods had a well-defined start and end, after which a new method followed subsequently. Some methods stretch over the whole project or were cycled several times.

4.1 Character Profiles

The profiles were progressively tailored by discussions about users, observations from meetings and later from the test results of the first user study. The users' own comments on their way of thinking helped a lot to nuance the picture of the characters.

To help the shaping process all the data was gathered to a bulk and written down in different categories; "experienced user" and "inexperienced user" as in a two-column *affinity diagram*. There was no need for more dividers because this was the only important distinction between the characters, as the domain knowledge seemed to be generally the same.

After a while two complimentary characters began to take shape from the categories. Whenever a new observation was made it was added to one of the two.

4.2 Design Decisions

This section aims to summarize all the important design decisions that had to be dealt with during the transformation phase. They are all strictly related to the usability of the system hence technical issues are not given account for. This section will only *issue* the problems and describe the reasoning behind the different views on each of them. In other words the *conclusions* regarding the decisions are left for the result chapter.

4.2.1 Hierarchical Structure

In order for the user to have a clear understanding of how to work with the page, he has to have a fair mental model of its architecture. If the structure is built with care the user will almost subconsciously find his way to the goal, whereas if the structure was poorly put together or mediated, the user would find it awkward and frustrating to navigate.

The architectural structure for sorting information is hierarchical as a direct consequence of having categories contain subcategories. Thus there wasn't a discussion whether *if* the structure would be a hierarchical or not, but how to present it to the user in the best possible manner.

Research findings indicate that users may give up if they are required to go through more than four or five clicks to get to the desired content (Rosenfeld '02). Too many choices can frustrate users by overwhelming human cognitive limits. A classic research article states that humans process information in chunks of 5-9 items (7+-2) (Lazar J)

4.2.2 Menu

The menu is an essential part of the applications usability; use a menu too simple and the user will get lost in the system, use a menu too complex and the user will get lost in the menu.

Tree node diagrams (e.g. Windows file explorer, see Figure 7) are great for handling hierarchical categories that contain large arbitrary information, because they are fairly compact and at the same time reveal the tree structure for the user.

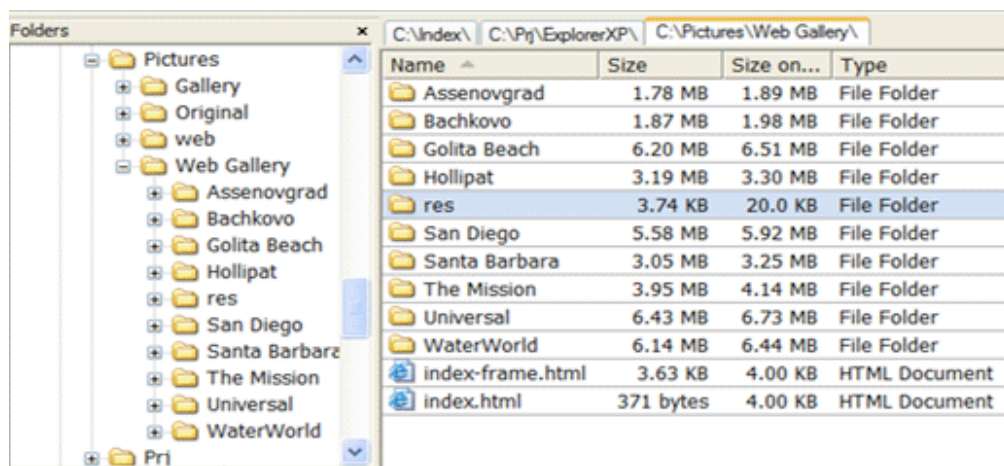


Figure 7, Windows File Explorer

Unfortunately there is a big usability loss in using tree controls because they are highly problematic for inexperienced users. Another drawback is that data can't always be categorized in a tree hierarchy.

We have seen countless of interfaces where programmers have forced non-hierarchical data into a tree control with the rationale that trees are "intuitive". While they certainly are intuitive for programmers (and other people are certainly becoming more accustomed to them), the big problem is that they do not allow users to capitalize on other, more interesting relationships between objects other than a strict hierarchy.
(Cooper A)

Another important aspect that affects this alternative negatively is that the model of a tree doesn't always provide a good visual representation of the users mental model.

Sometimes a *nested set*, also known as *tree map* is a more appropriate visual representation (see Figure 8). Unfortunately a nested set is too space consuming and not a good option for browsing, hence makes it inappropriate for navigation purposes.

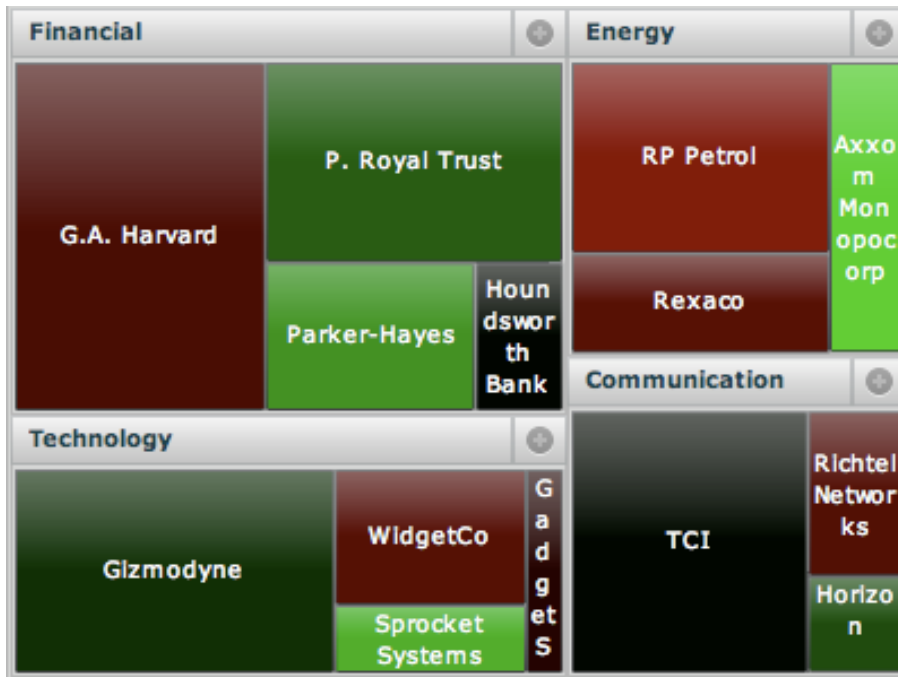


Figure 8, Tree map (taken from blogs.oracle.com/xmlpublisher)

4.2.3 Breadcrumbs

Lots of web pages with deep and complicated node structures make use of *breadcrumbs* in order to give the user a landmark of where he's at. The term "breadcrumbs" comes from the fairytale *Hans and Gretel* in which they leave a trail of breadcrumbs when going into the forest in order to be able to find their way out. Breadcrumbs in a website typically look like figure 7.



Figure 9, Breadcrumbs

Accordingly, the node path of which the user has navigated down the tree structure is plotted from left to right with each subsection being a hypertext link to its level. If the user wants to traverse the tree backwards he can do this by using the links in the breadcrumb.

4.2.4 Calendar

A very central part of Doctum is the calendar. Much of the information in the system is time dependent:

- *Class attendance*: the class occasion is connected to the date and time on which it is scheduled
- *Event*: An event is relevant put in perspective to on what date it occurs
- *Exam/Test*: always occurs on a specific date.

The calendar has other positive features such as being a mainstay to remember today's date, how many days the current month has, as well as giving an overview of when events occur.

It was decided that the calendar should be visible to the user in the right column permanently. An objection to this choice is that the calendar shouldn't be visible where it's not relevant to the content in the main field. This is a perfect con; information that is irrelevant could be both confusing to the user and space consuming.

However the alternative would be to have the system switch between different views depending on what content the user is browsing which could possibly be even more confusing to the user than showing the calendar at all time, thus discarded.

Because the calendar would take some time to be implemented and it didn't have a critical function in the attendance view it was decided to leave it until after the second user test. Thus the calendar is not further explained in the result chapter.

4.2.5 Save Button

An important consistency issue was regarding whether if using a *save button* in e.g. views such as the attendance view was a good choice of design. Traditionally forms in web pages are confirmed with a save or a send button. While there are positive sides of choosing the option that is norm (because the users are more familiar with that way of performing an action), it's important to be able to criticise old conventions when they are not filling a need or are even negative to the design.

If you ask users themselves, they will reject the new solution because they abhor change, particularly when that change affects something they have already worked hard to master - Like the file system. However, users are not always the best predictors of design successes, especially when the designs are different from anything they've already experienced.
(Cooper A)

There are different ways of reasoning about the save button; The Ajax script could be turned off and let the changes only be stored locally until the user clicks the save button. This would assist the user in knowing when the information is saved on the server and provide a receipt that the information is actually sent.

A potentially positive (but yet unproven) consequence of the fact that the information stays on the client until the user manually saves it is that it could have a calming effect on how the user interacts with the system. The user may be more casual in terms of e.g. what comments he writes about a student in the attendance form because he knows it is not made public before he actually saves it.

It is also easy to see the functional benefits with auto save. Take for example the following scenario: A teacher is logging her students' attendances at home. When she is half way done her cat chews through the PC power cord and all the browser data in the RAM gets lost. When she starts up the computer again and logs in to Doctum, the changes she had made before the PC shutdown had been saved on the server so she can continue from where she was.

A solution to both previously stated questions could be to implement a rich feedback system that notifies the user when information is saved and when the client experiences connection problems with the server.

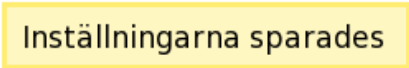
4.2.6 Feedback

Suggest that the system automatically saves changes to a page, the question then becomes "how will the user know that his data is saved?" the information could of course be sent to the server invisibly without making a noise. The user would then, after a

while, learn that changes were automatically saved. But what if the client loses its connection to the wireless network during the process? And also, “Since most users are used to a manual confirmation of data, how would this solution incorporate with their mental model?”

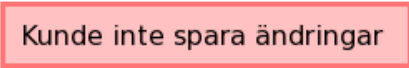
There are many ways of providing feedback to the user. Naturally an unobtrusive modeless feedback should be used so that the user doesn’t get interrupted in his workflow.

People are very perceptive to changes in hue, which makes it an efficient tool for catching the users attention. Extra cautiousness has to be taken when alerting users. The notification has to be of relevant nature and can’t be used too often; otherwise the user will filter it away. Figure 10 and Figure 11 show how colour notifications could be used to alert the user about how the data is received and handled by the server.



Inställningarna sparades

Figure 10, Moderate notification “The changes were saved”



Kunde inte spara ändringar

Figure 11, Critical notification “The changes couldn’t be saved”

The yellow colour is generally perceived as of lower importance level, whereas the red makes the user aware of that something is faulty. Thus, a yellow notifier can be displayed without the user making a note about it. If he wants to make sure the data was saved he will just look for the yellow tile. If the red tile appears in the users field of vision it should draw his attention to it, and he could adapt his actions to it or take other precautions.

4.2.7 Colours

The choice of colours is a very important aspect of the interface; they should help the interface look harmonious and at the same time as they have to convey the right expression of what the program is for. Doctum is intended to be used once or twice every day by its users, therefore the colours can’t be too vivid and intense which would cause annoyance after a while.

The colour theme that was chosen is brightened shades of blue. The reason of which is the plain hue that is pleasant to watch, plus it gives an informal expression.

4.2.8 Multiple Views

Since information from different parts of the system (modules) might be relevant to each other when working with a specific task, the alternative of using *multiple views* in the system was considered. The views could be alternated between using a *tab interface*.

By using this system the user could have many tasks running in different views simultaneously from which the effectiveness could benefit if the user utilized it properly.

Ultimately it was decided not to use this feature because it would make the system more complicated for users that weren’t used to switching between views. Also most browsers has built in tab support with which users that *want* can run multiple views.

4.3 Paper Prototype and Storyboard

During the first stage of the design phase a *paper prototype*, also called a *wireframe* of the attendance page was constructed. The prototype wasn't really made out of paper, instead *Balsamiq Mockups* (www.balsamiq.com) a graphic library of standard web components, was used.

The interface posed several important questions to be dealt with. It had to be easy to understand, extendible, fast to navigate, yet be consistent and not bother already initiated users with superfluous information. Balsamiq makes it simple to construct wireframes for sketching basically any standard web page. Using this tool was very useful because it made the process of drawing the prototypes and sharing of them more effective. It saved a lot of time for the project.

Storyboards are a little messier than paper prototypes; they have to be highly scalable to fit in a wide range of system structure and at the same time be able to support intricate detail descriptions. There are several good storyboard programs, which could have been used, but in the end the simplicity benefit of traditional pen and paper won over the technical features, which these programs have to offer.

4.4 User Study

There were several months between the first and the second user test. The main reason to have two separate tests was to have plenty of time to evaluate and implement the changes that the users proposed in the first test before starting with the second.

The idea was to have a broad perspective in the first user test (testing architecture, orientation etc.) and then narrow the scope down in the second (focusing on details such as slider button, comment field etc.)

4.4.1 First User Test

The test was carried out with a think aloud test followed by an interview. The think aloud test was conducted with a set of tasks, which the user was asked to perform for the first time. The tasks were simple standard routines that the user should be doing frequently when the system is put in use.

4.4.1.1 System Description

The test system was consisting of a *log in page*, a *user home*, *reminder view*, *attendance view* and a *your subjects view*.

Logged in the user was put in to the *user home*. The alternatives were quite few, either he could go to the *your subjects* page or to the *reminder view* through clicking the *attendance* link in the navigation menu. From there the user could click on a class occasion to get to the *attendance view* where the students in that class were listed with pagination.

4.4.1.2 Test Description

The think aloud test had the following four areas in focus:

- System architecture/model,
- Links/link names,
- The view for setting attendance
- The reminder view

A list of tasks, which the test person should perform, was constructed, it had the following tasks:

- Use the system to see what subjects you are teaching
- Try to find out which was the last elapsed lecture
- Go to the set attendance page
- Mark the last student in the list as “away” (test user had to use pagination)
- Go to your home
- Log out

4.4.1.3 Evaluation

Meetings were appointed and affirmed with seven teachers, the expected *no show* was 15%, which would leave about six test interviews. Sadly the no show rate proved to be rather 60%, leaving three users, which was three too short. With some re-appointments six interviews managed to be completed when the time plan for the test was much over due.

Although tests were expected to take at most 40 minutes it leaned towards 50 minutes in most occasions, mostly because of that the interviewee was interested in discussing other aspects of the system, which were not covered in the test.

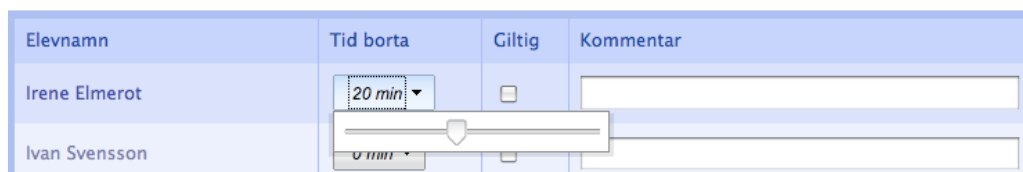
Only one person was conducting the tests. The think aloud tests could sometimes go too fast when there had to be kept notes about what the test subject said and at the same time what he was doing in the system. This might also be a reason to why the tests took longer than expected. Sometimes the interviewee was asked to stop for the conductor to catch up with writing the notes. Optimally think aloud tests should be conducted by one person handling the communication with the user and another person observing the user and taking notes.

4.4.2 Second User Test

When the system had the functionality and a GUI for the page where the user logs student attendance the second test was conducted. This test was a bit delayed because the functionality behind the slider buttons took more time than it was planned for, also the feedback system had to be re-designed several times because of technical issues with AJAX.

4.4.2.1 System Description

In addition to the functions that was in the first user test a few extra features had been added (see Appendix B). In the *set attendance view* there was a slider button with which the user could select the time, which a student had been absent, a checkbox for if the absence was valid (i.e. the parents had authorized it) and a commentary field, see Figure 12.



Elevnamn	Tid borta	Giltig	Kommentar
Irene Elmerot	20 min	<input type="checkbox"/>	
Ivan Svensson	0 min	<input type="checkbox"/>	

Figure 12, Set Attendance View

There was a feedback system (a modeless overlay box showed in Figure 17 in the result chapter), which was intended to inform the user when the changes he made had been

saved to the server and when they couldn't be saved because of connection problems with the server.

The *reminder view* had been updated with two *quick buttons* to make the users normal use cases more effective, see Figure 13, Reminder View

Ämne	Årskurs	Datum och Tid		
Engelska C	3	2009-05-26 20:14:53.0	Alla närvarande	Lektion inställd

Figure 13, Reminder View

4.4.2.2 Test Description

The test users were given a personal account, which was valid for two weeks. The users were selected from the upper classes in the school where the teachers are familiar with setting attendance on their students.

The users were asked to use Doctum at any time they would normally use the system if it was finished already, meaning both in stressful situations (like during a class) and in relaxing environment (e.g. at home). They were also asked to keep notes meanwhile they tested it in order to fill in a questionnaire that was sent with the account details.

The areas that were in focus for the test were:

- Slider button
- Comment field
- Feedback system
- Save button
- Quick buttons

4.4.2.3 Evaluation

There was a big interest in testing the system when the system was presented for the test group, but when the test period actually started the vivid curiosity seemed to have dried up slightly, so when the test was supposed to be finished many of the users hadn't actually logged in to the system. This naturally delayed the test for one week further, and gave poorer results because the last test subjects to answer the questionnaire were slightly unmotivated.

5 RESULT

In this chapter graphical output of methods as well as conclusions are presented and analyzed. Commentaries are only explanative

5.1 Character Profiles

Early in the project two character profiles were created in order to help reasoning about different users' needs. They were designed to be the complete opposite each other. One of them being basically a computer illiterate with little patience with computers, and the other being a computer geek with a strong will to improve effectiveness with technology.

Here follows a short description of the profiles:

- “Grete, The Inexperienced computer user”, gets irritated when the system is too complicated. She is a ragged teacher who has been in the profession for 30 years and is used to be on top of things. When she started out as a young teacher there weren't any computers around. She was perfectly happy with doing everything from grading to logging students attendance on paper and can't see a reason why she can't do that still. She has a short temper and computers gets on her nerves if they complicate her work, hence satisfying her demand a very pedagogic, easy to use interface with a clear provable of benefit.
- “Hans, The Experienced computer user”, gets frustrated when things take too much time and slows down his work. Hans is a newly employed teacher and has had a great use of computers during his studies. Being very passionate about his work he has a very pragmatic attitude towards tools he uses in it. He spends a lot of time towards the computer at his home and knows his way around. If a program takes too much time to learn he lays it off and even more disturbing are programs that aren't effective enough. Satisfying Hans must be achieved with a powerful, easy to learn program that is scalable to fit his needs.

The character profiles were a great profit when discussing usability aspects. Especially when the two characters' needs conflicted with each other they were helpful to evoke a compromise between the both. Without them an easier solution may have been sought.

5.2 Design Decisions

Here the result of the design decisions, which are posted in the previous chapter, will be showed and explained.

5.2.1 Hierarchical Structure

There is an important balance between structuring the pages with too many levels (hence deepening the tree), and structuring it too shallow (which creates a wide array of links in the menu) In order for an interface like this to be successful the hierarchy

architecture had to be kept simple, i.e. the node diagram (see Figure 14) could not be too deep (a soft cap was set to three levels) because otherwise the user could be uncertain of in which node he was browsing, hence it would cause confusion.

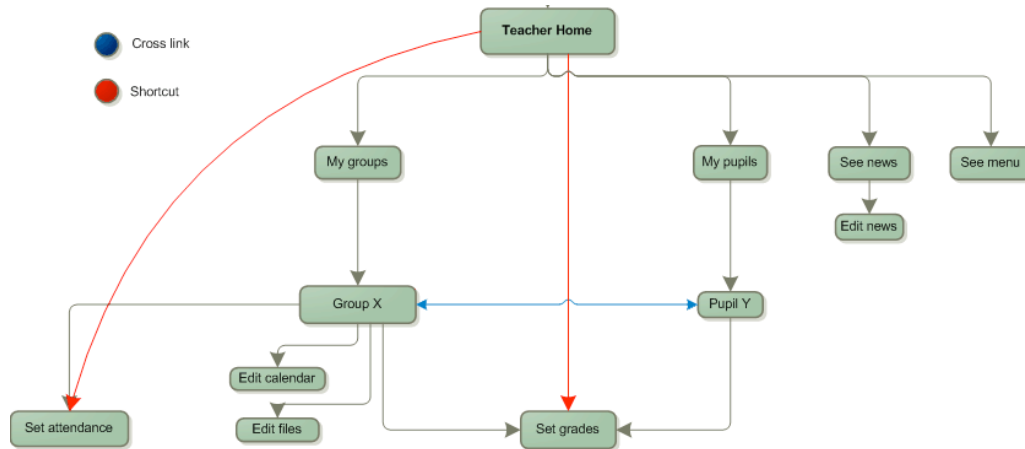


Figure 14, The Hierarchical Structure

In the design sketch three things should be noted in particular:

1. The 2nd level nodes in the tree are available from the main menu in the teachers home.
2. In addition the shortcuts marked by red arrows are available from the menu. The reason for this is because they are the two most frequently used cases and thus should be easy to access, “If possible, consider giving users zero-click access to the answers, meaning that the homepage displays the most-needed information automatically.” (J. Nielsen)
3. There is a *cross link* marked by a double arrowed blue line between two branches of the tree; from group x it should be possible to list every student in that group, and mutually looking at a pupil there is a direct link to in which groups he/she is registered.

Figure 14 is an early sketch from the design phase that has been changed several times afterwards.

5.2.2 Menu

The dilemma (explained in the section *Menu* in the method chapter) was how to display a -at some places complex -tree structure for the user without losing usability by making it too complicated to the user.

It was decided to *present* the categories as a plain structure but keeping the hierarchy structure underneath. This solution gives the system technical benefits by being able to store each category as an ordered set in a tree hierarchy. But most importantly it enforces usability because the users mental model of the page is subconsciously coherent with the hierarchical structure. By using this model the architecture could be extensible

and robust and at the same time the navigation could be made clean and simple.

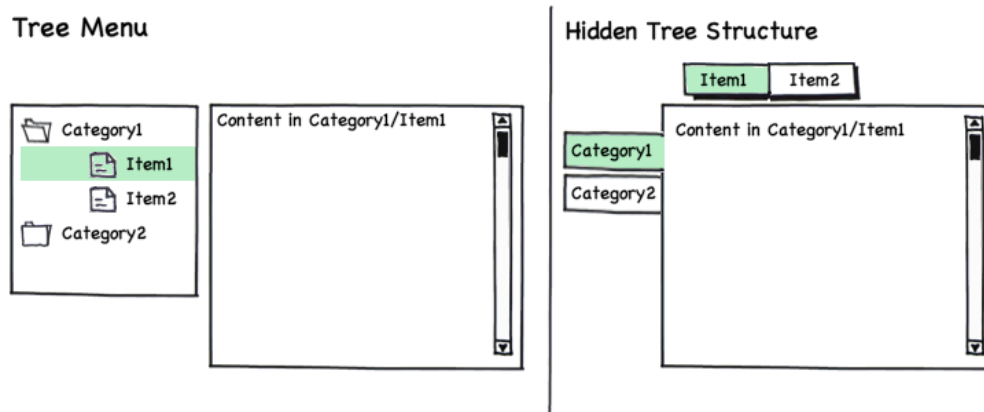


Figure 15, Different Menus

Figure 15 shows the different ways of presenting a tree structure. Figure 16 shows how it was realized in Doctum. The only difference is that the name of the subcategory is displayed as a header over the table instead of like a tab.

Frånvaro

[Tidigare hanterade lektioner](#) | [Frånvarorapport](#)

Lektioner som du ännu inte satt frånvaro på

[Hem](#)

- [Frånvaro](#)

[Ämnen](#)

Ämne	Årskurs	Datum och Tid
Engelska C	3	2009-05-26 20:14:53.0

Figure 16, Hidden tree structure in Doctum

The biggest loss with using a hidden structure is that if the tree is more than two levels deep, the location in the hierarchy has to be displayed with *breadcrumbs* or in an alternative way.

5.2.3 Breadcrumbs

Ultimately the idea of using breadcrumbs was discarded, mainly due to the fact that it would add information that a majority of the users would experience as superfluous or even annoying in some cases. In the article *Breadcrumb Navigation: Further Investigation of Usage* the authors' present figures from a survey that showed 40 percent uses breadcrumbs for navigation.

Breadcrumb users were found to use the Back button less often than users who did not use the breadcrumb; however, no differences were found in the efficiency measures of total pages visited, navigation bar clicks, embedded link clicks, or time to complete the search tasks. It is not known if all participants understood the function of the breadcrumb as a navigational tool. (Lida B, and Chaparro B)

Breadcrumbs have other problems directly inherited from the tree structure; not all information is fit for being put in a hierarchical structure.

To use path analysis effectively, the web site must be organized hierarchically, with only one specific path to reach a certain web page. For instance, if users could take 10 different paths to reach a certain web page, then it could be hard to provide a trail of how they reached the page, If users could reach the Leadership Program page under Campus Life, Academics, Libraries, Faculty and Staff, and so on, what path would be presented? It could be confusing, since the user would read the path and say, “but that’s not how I got here!”
(Lazar J)

Since Doctum needed to make use of *cross-links* between tree branches this issue becomes a big problem. An alternative would be to have a chimera tree that is only used to support breadcrumbs, but since there would be a discrepancy between the users way of navigating down a branch and the chimera tree it would loose some of its purpose (as a page history) to the user. Also it would reinforce the tree node structure, which had been decided to be kept hidden from the user.

5.2.4 Save Button

Because of the unnecessary excise it was decided to not have save buttons where the user makes changes. If the information is sent to the server streaming and consistent this will be both faster and less stressful for the user (he wont have to remind himself if he actually saved the changes before he closed the system).

The reasoning to why the auto save was chosen is discussed in further detail in the discussion chapter because analysing its usability aspects it is one of the goals for this thesis.

5.2.5 Feedback System

The implementation of the feedback system was quite straightforward. It followed the design sketches quite rigorously except on one point; the feedback overlay was sketched to be in the top of the browser window in such way that if the user scrolled down/up the overlay box would follow after (staying on the top).

Unfortunately due to technical problems with YUI’s layer boxes it wasn’t possible to make the overlay stay in the same place when the window was scrolled so an alternative solution was invented.

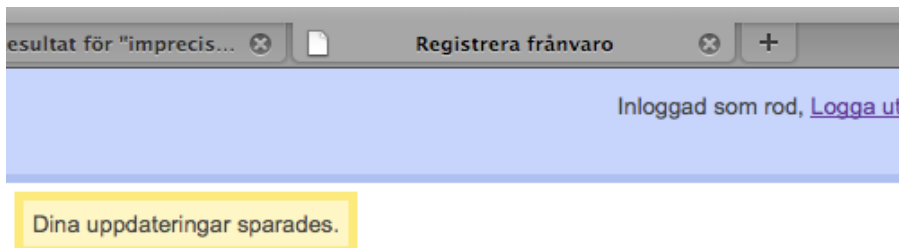


Figure 17, Feedback “changes saved”

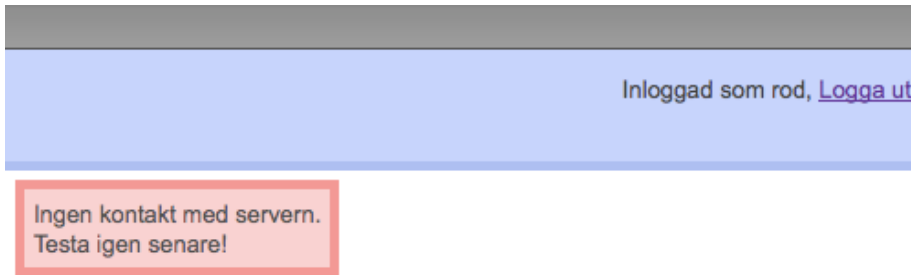


Figure 18, Feedback “error”

The overlay was put in a fixed position between the header and the body content (seen in Figure 17 and Figure 18). This way the feedback would be in center of focus when the user is browsing the top of the page. Obviously, if the user would scroll down on the page the feedback wouldn't be visible, but the user could always scroll up to make sure that a piece of information was saved.

5.3 Paper Prototype and Storyboard

The output of the mock-up method resulted in a set of sketches (see Figure 19 for an example) of the different pages in the system. From these sketches it was possible to imagine how it would be to navigate the system through the menu to the left.

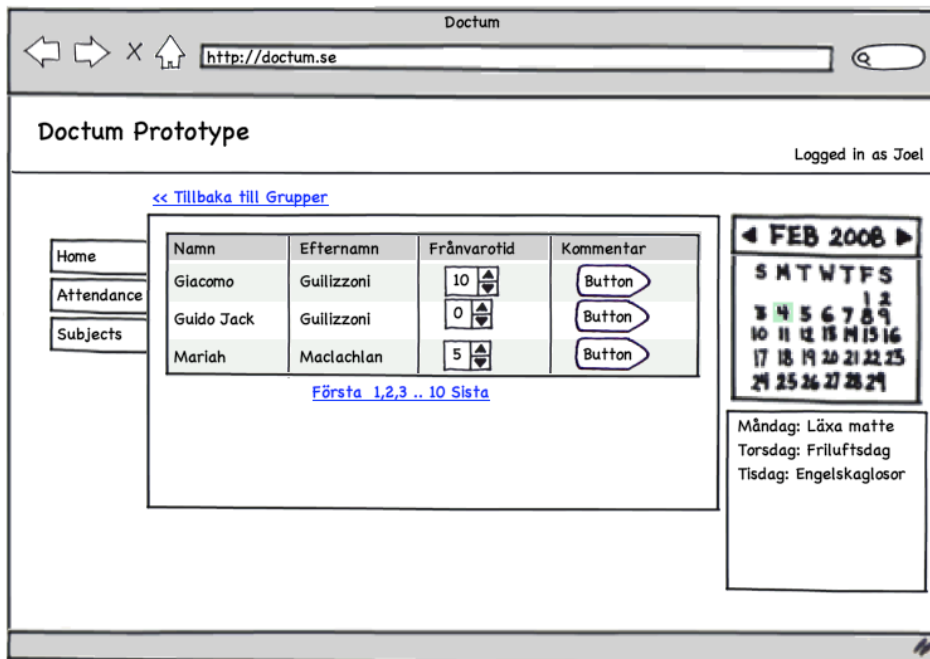


Figure 19, Paper Prototype

The paper prototype was very simplistic but contained all the elementary features that the page was supposed to carry. Foremost it served as a mental model, which helped the later design of the system. It also revealed some initial design flaws such as the amount of information that was possible to fit into the page.

For the GUI to be able to carry all the information and still be structured to work with it was decided to use a view with three columns (see Figure 20) plus header and footer because this would profit the large amount of data that should be possible to be displayed concurrently.

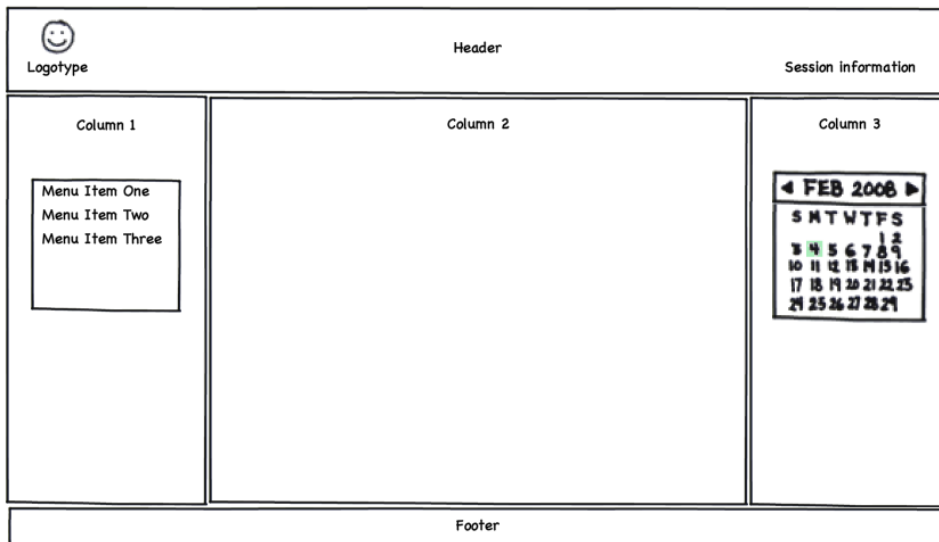


Figure 20, Three-column layout

According to *Usability checklist* (2004) the logotype was placed in the upper left corner of the header and the session information was put in the upper right. In the three-column row the menu is put in the leftmost column (column 1), the calendar in the right column (column 3) and the column in the middle (column 2) is generally filled up with the content that the user is browsing. Column 3 is of dynamic width so its content adjusts to the screen size.

5.4 User Study

The two tests that were conducted resulted in fragmentary sentences, quotes and observations. The gathered result (found in the appendix) from the notes has been shortened and compiled into readable form in this chapter.

5.4.1 First User Test

This test took 40 minutes and was constructed in two stages; the first half was a think aloud test and the second was an interview.

5.4.1.1 System Architecture

The test subjects seemed to be able to grasp the model of the system quite quickly. Everyone realized that the left menu was the way to browse the different sections. One test subject seemed a bit confused when he wanted to go up a level in the tree and was asking why the breadcrumbs weren't clickable (there was just a breadcrumb text). The others weren't looking for this kind of navigation feature at all.

5.4.1.2 Reminder View

One user was a bit confused with the header "Lectures you haven't logged" ("Lektioner du ännu inte satt frånvaro på"). "-Is a group the same as a lecture?"

Some of them used the list sort to order by dates (it was randomly distributed by default). Many of them were confused with the way the occasion's time is presented (yyyy-mm-dd, hh:mm) "It would be better if it said what day of the week the lecture takes place".

One pointed out that they normally don't have more than one or two lectures in a subject in a class per week so the text could read "Mathematics with class 9, Friday at 9:00". Also the week number should be displayed in the row.

A few of them said that it could be stressful with *reminders* stacking up, and it has a negative sound, but one of them saw it as a positive incentive "It's psychologically stimulating to keep the list short".

5.4.1.3 Orientation and Navigation

When asked to perform the different tasks, everyone seemed to find the best way without longer detours or much thought, which has to be interpreted as the link names and their positioning were well designed.

One test person said he wanted more icons to not have to read the links, aiming on the left menu. The others in the group seemed to have no problems with reading the links, and when they had to go through a page twice they remembered instinctively where that link was.

5.4.1.4 Attendance View

Most of the test persons that had worked with the old system *B*. were enthusiastic about the amount of time it could save and they seemed happy that this system knows by default how long a lecture is "-sometimes when you are filling in for a colleague you don't know how long it was"

One test subject couldn't find/use the pagination to browse to the end of the class. When instructed of how to do it he seemed to understand how it worked quite well. The other test persons were familiar with the way of navigating lists and were instead talking about what functions that should be in the view. "There should be a text field for writing comments!" or "I would like a list of pre-set reasons for why the student is away".

"Clicking on the student names should take you right to their attendance statistics" was a common opinion among test subjects. Many of them tried to click on the students' names even though they weren't displayed as links.

They all point out that there should be a way to set the amount of time the student has been away, not just present/not present.

5.4.2 Second User Test

For this test the users were given access to a private account to the test system and got to try it for two weeks. Along with the account details they were given a questionnaire, which they could fill in meanwhile they were working with the application.

5.4.2.1 Slider Button

All the test users agreed that the slider button was a good solution to the task of choosing absent time amount. There was some expected criticism, which wasn't submitted, such as span of the ticks (five minutes) on the slider. Someone might say that five minutes is too imprecise while someone else may think that an exactness of ten minutes would be sufficient.

5.4.2.2 Comment Field

The comment field was the most criticised object in the attendance view, basically because it didn't allow more than 200 letters about each student per course occasion. The main reason behind this cap was that the intention with the field was for the teachers to write *short note* about e.g. why a student was late whereas most of the test users wanted to use the comment field for writing longer comments about e.g. the student's behaviour in class.

5.4.2.3 Feedback system

As explained in the realization chapter the mock-up system had a problem with that the feedback overlay box wasn't visible if the user scrolled a bit down on the page.

The user test showed that many of the users had completely missed the feedback box, probably because it had been outside the boundaries of the window. Upon the question "Was the yellow feedback box helpful?" two of the test users answered "What yellow box?" the same users were concerned about if the information was ever saved.

The lesson to be learned from this is probably that the feedback must be visible *everywhere* on the page; otherwise it only causes confusion among the users.

5.4.2.4 Save Button or Auto save

The system that was used previously on the school had some technical flaws regarding the saving of data; it didn't have a save button and in order to save something the user had to click somewhere outside the field that was edited, something that was often forgotten by the users and caused much annoyance.

Naturally the users that were disappointed with the old system found the absence of a save button in Doctum a bit disturbing. The complications with the feedback system made their experience even more awkward. But among the users that had noticed the feedback box and had realized that they didn't have to click outside the area to get it to save it was seen as a good feature.

Since the auto save was very much associated with the old system's shortcomings the resistance to this feature in particular should probably be looked upon with a grain of salt which some of the users were quick to mention themselves.

There were some who had noticed the feedback box also. Their response to the auto save feature were much more positive. One test user had tried to cheat the system and after a while realized that it wasn't possible and from thereon gotten very enthusiastic about it.

5.4.2.5 Quick Buttons

After the first user test two *quick buttons* were added in the reminder view; "all present" and "lecture cancelled". If the former button is clicked then the lecture is added to the statistics with all the students counted as present and the latter just erases the lecture so that it doesn't affect the statistics.

Both the buttons were widely appreciated among the users, generally because they made their work easier. There occurred questions about what the buttons actually did; if "lecture cancelled" really deleted all trace of the lecture or if it still had a statistical effect. One good way to smother this kind of uncertainties is to use *tool tips* on buttons.

5.5 Final system description

After the two user tests the system had gone through several stages of refinement. The system architecture had been modified slightly from the first storyboards (see Figure 21)

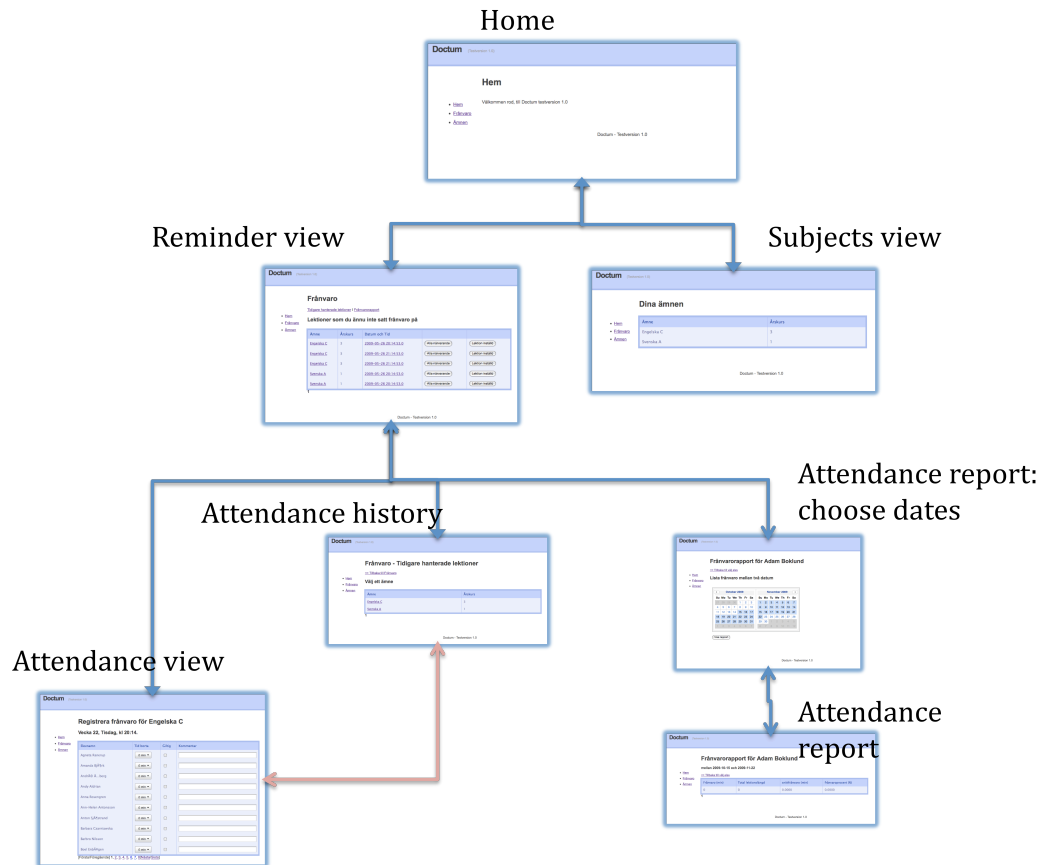


Figure 21, System architecture

From the *home* view there are two alternatives; *subjects view* or *reminder view*. The subjects view simply lists the subjects, which the user teaches in. Whereas the reminder view has a few more functions (see figure 22)



Figure 22, Reminder view

By clicking on a subject in the reminder view the user is taken directly to the attendance view. The screenshot also shows the two *quick buttons*, with which the user can register all students as present on a class occasion (by clicking the *all present button*) or cancel an occasion (by clicking the *cancel button*).

In the attendance view (see figure 23) the user can register attendance for all the students in that class and also write a comment about students.



Figure 23, Attendance view

By clicking *previously handled lectures* in the reminder view, the user is taken to the *attendance history* from where it is possible to browse all the previously treated classes.

The link *report* in the reminder view takes the user to the *attendance report view* (see figure 21) in which the user can, by selecting a student and a time span, see an attendance report for any student.

6 DISCUSSION

This chapter will discuss how the application turned out with the result from previous chapter and how well it met its goal. It will also be discussed the verifiable goals of the project.

When the second user study was finished the attendance module for Doctum was still a long way from completion. Important pieces were still missing, e.g. AJAX tables and a proper attendance report for printouts.

However the system went from basically just an empty framework to a fully functional web application. During the second user test many of the users felt encouraged to start working with it immediately, which -though not scientifically -is a good grade for the usability's level of completion.

It is difficult to say anything from an evidential point view about the usability by judging from the test results, because both of the tests were made with small user groups. Thus it is impossible to verify if the chosen solution (e.g. "the save button") was better than its alternatives. In the end it was just the users' *feelings* that were weighted together combined with a coherent logic that decided which alternatives were chosen and which were discarded.

6.1 User Study

The lack of statistical data could of course serve as critique against the interview oriented surveys that were conducted. Indeed it would be a great method to verify the success/failure of the decisions made during the design process but the general problem with conducting a statistical survey is that it wouldn't be *forward oriented*; a questionnaire with close-ended questions could have given very interesting stats on an already existing system and could, with proper evaluation, prove some points about the systems usability, but it wouldn't have given a clue about how to invent the design in the parts of the system that were going to be designed next.

Accordingly, since there weren't enough resources to conduct an exhaustive statistical survey and because the system wasn't in one of the final levels of completion, the interview method was chosen *partly* in order to give feedback on existing functions, but *mainly* to give guidance about how to actually design the remaining parts of the system.

The first test had a lot of engagement from the users. There was no problem to get six users to spare the forty minutes, which was the approximated test time. The interviewees were eager to discuss a lot of ideas and even mailed in some comments afterwards.

In the second user test the interest had for some reason become smaller although almost all test users from the first test were exchanged to new ones. Perhaps the reason was not because of the system in itself but in how the test was conducted. In the first test the system was not tried before and was completely new for the users that participated

in the test. In the second test the users were given a log in account, which was valid for two weeks.

It would be reasonable to think that it would be easier to get nuanced test results from the second test where the users were given the freedom to try it at home in their own pace. However, during the two weeks the test users were only logged in the first few days of the two weeks and the very last.

It could be interpreted like when the test user first logged in the system was new and exciting. After the curiosity had been saturated the user logged out of the system, thinking: "I'll fill in the questions later, I've got two weeks to do it!" And after that the task was forgotten about until the last day when only a few remembered to return the questionnaire.

There could be a lesson in this; More time isn't synonymous with more test answers, it's rather sometimes the opposite. It's better to put the effort in to schedule meetings with the test users than to chase them after questionnaires.

6.2 Verifiable goals

Here the goals will be discussed based on the facts found in the literature study and the results from the conducted tests in this thesis.

The question formulated in the beginning of the report is "How can a web application for logging students' attendance be designed to support usability?" What the question aims to investigate is; which are the general guidelines that one can learn from this project to be applicable to similar system designed in the future.

The test results showed that the most important usability aspect to think of in order to satisfy the users was -perhaps not surprisingly -*effectiveness*. Spelled out, the program should help its users to achieve their tasks fast and accurately and with a learning period kept to minimum.

This is of course what could be called a *general guideline* but it should be especially considered in the kind of applications to which the users don't spend most of their day sitting in front of a computer but rather the opposite; they log in to the computer for about an hour before they go off work in the evening and then the saving of data is the only thing that lies between them and going home. The circumstances make the users extremely impatient with the program and this raises the demand on effectiveness to a higher degree than in a general application.

The second question was "Can *auto save* be used instead of a save button to enforce usability?"

The literature study was ineffectual on this point, probably because it's a very new area especially for web applications. There is a lively discussion about auto save in the blogosphere though, but the views seem to differ widely. However, an opinion which seem to pop up in this discussion is that the save button is in fact unnecessary excise and belong to an earlier computer era;

It's invented by engineers to move data memory to disk back in the days where tape and disk space was incredibly expensive (and time consuming): Many computers didn't even have disk or tape. There was an extra need to actively decide to get that delay it took to save stuff. Nowadays, some smart web applications offer automatic save (for example GMail and Writely). The next step will be invisible save, and we'll probably end up with no save button as implemented by Palm Pilot. (Usability blogg, 2008)

The auto save feature was one of the main focus areas in the second user test. The test showed one thing in particular; *without* a visible feedback system the auto save was generally disliked, but on the opposite side for the users that trusted the feedback system the auto save was considered a good improvement.

The test proved that the auto save is big a step towards increased usability for web applications, but the world is not yet ready for invisible save because the “save button” is so deeply rooted in people’s consciousness.

With the results from the test the usability can be discussed theoretically from the basis the ISO definition of the term: usability is the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use (described further in the theory chapter of this thesis) The auto save obviously increases the efficiency because the user doesn’t have to be bothered with the unnecessary excise of clicking the save button when his changes are finished. The effectiveness can be helped somewhat like for example when the data is saved on the server when there is a power failure on the users PC. Finally when the feedback system is implemented properly the test proved that the user satisfaction was increased.

All in all the auto save feature should be taken into consideration when building any kind of application in which the user has to edit a lot of data, but which doesn’t put too heavy load on the band width and the server.

6.3 Conclusion

The application turned out to be a whole lot different than what it was expected to be; A lot of functions that were planned on getting implemented just didn’t make sense after that e.g. a paper prototype had been created, or because of something a test user said or been observed doing, in the first user test.

Similarly but in the opposite direction there were new ideas, which were born from methods such as these. One of the main areas of discussion in the initial part of the design was how to combine multiple views e.g. a teacher might want to have one view with the students attendance in a course visible while he sets the grade for the student in the same course. Paper prototyping showed that even though this was highly feasible, it wouldn’t be easy to understand for new users especially if they were forced in to using multiple views.

After the first user test multiple views were even more obscured, because this showed clearly how the users were used with working in programs. More views would just have given cause to a lot of headaches. Also it showed how the test users that were experienced with computers could handle many tabs in a web browser to quickly alternate between views that were relevant to a specific task.

In that fashion ideas were processed during the whole project. An important lesson from this is perhaps to never stop elaborating with new ideas. If the design process would have been shortened many of the good ideas, which were invented in the later stages of the project, wouldn’t have been implemented and many of the bad ideas would have stayed.

References

Chris Jones, J (1992) *Design Methods*, Second edition with new prefaces and additional texts. John Wiley & Sons, Inc. New York.

Cooper, A., Reimann, R., Cronin, D. (2007) *About Face 3, The Essentials of Interaction Design*. Wiley Publishing Inc. Indianapolis.

Hippel (1986) *Lead Users: A Source of Novel Product Concepts*.

Jan P. L. Schoormans et al, Roland J. Ortt, and Cees J. P. M. de Bont (1995) *Enhancing Concept Test Validity by Using Expert Consumers*.
Blackwell Publishing Journal of Product Innovation Management, Volume 12, Number 2.

Lazar, J (2006) *Web Usability, A user-Centered Design Approach*.
Pearson Education Inc. New York.

Lida, B., and Rogers and Chaparro, B (2003) *Breadcrumb Navigation: Further Investigation of Usage*. Usability News, August 2003, Vol. 5 Issue 2.

Nielsen, J., and Marie Tahir (2001) *Homepage Usability: 50 Websites Deconstructed*. New Riders Publishing, Indianapolis.

Nielsen, J., and Tahir, M. (2001). *Building Web Sites With Depth*.
WebTechniques, February.

Roughley, I (2007) *Practical Apache Struts2 Web 2.0 Projects*.
Apress, New York.

Snyder, C (2003) *Paper prototyping, The fast and easy way to design and refine user interfaces*. Morgan Kaufmann Publishers, San Francisco.

Internet references

ISO9241-11 3.Definitions(1998)
<http://www.fd.utokai.ac.jp/~nosu/english/define.html>

Usability blogg (2006)
<http://justaddwater.dk/2006/11/27/save-button-usability-issues/>

Usability checklist (2004)
http://www.netmechanic.com/news/vol7/design_no4.htm

Ten Step to Personas, Dr. Lene Nielsen,
<http://www.hceye.org/HCInsight-Nielsen.htm>

CSA

<http://www.csa.com/discoveryguides/scholarship/gloss.php>

Alertbox, Jacob Nielsen's (2000)

<http://www.useit.com/alertbox/20000319.html>

Appendix A

User test 1

Observations/Question Answers

Check what subjects you have as a user
<p>1. Hittar direkt ämneskategorin. Han tycker att ordet "kurser" passar bättre för gymnasium, ämne/ämneskod är bättre. Start/slutdatum är för honom irrelevant.</p> <p>2. Går direkt till ämnes-kategorin. Förstår vilka ämnen som tillhör honom. Tittar inte nämnvärt på datumen.</p> <p>3. Hittar direkt ämneskategorin. undrar över start/slutdatum men säger sen att han hade kopplat om datumen stämde med de vanliga tiderna. Kurskod använder han/de inte.</p> <p>4. Går direkt till ämnes-kategorin. Undrar vad datumen betyder. Försöker klicka på ämnet att få upp mer information om det.</p>
Try to figure out how many lectures in Swedish you haven't registered attendance for. (Is he/her aware of the sorting function of tables)
<p>1. Går till frånvaro. tittar enbart på det övre fältet "lektioner du inte satt frånvaro på". Studerar tid/datum och använder list-sorteringen för att försäkra sig om att datumen är korrekt ordnade. Hittar den senast avslöpta lektionen. När hon blir ombedd att kolla de lektioner inom svenska som inte är satta frånvaro på så räknar hon lektionerna i "lektioner du inte satt frånvaro på" istället för att gå in under kategorin "ämnesgrupper". Gillar utseendet och att lektionerna som inte är avklarade listas.</p> <p>2. Hittar frånvaro-kategorin snabbt. väljer en (fel) lektion i mitten under "lektioner du inte satt frånvaro på" och börjar trycka i checkboxar. Efter att han blivit uppmärksam på att han är inne på fel klass återgår han till föregående sida och studerar datumen på lektionerna i listan. Hittar lektionen. När han bli ombedd att sätta den sista eleven i klassen som frånvarande väljer han personen längst ner i första fältet. Han missar alltså pagination. Vid närmare koll hittar han dock möjligheten att stega.</p> <p>3. Hittar direkt frånvaro. Sorterar inte. -Är lektion en grupp? tycker att tiden är förvirrande vill bara se datum och tid. saknar vecka! vill kunna sortera efter vecka. någon slags färg-tid-skala hade varit uppskattat. Undrar om det inte går att se gammal frånvaro. Vill kunna se detta per elev v. & mån i kalendervy. Gillar tanken att försäldrar ska kunna se och bekräfta att de gjort så via hemsidan. ändrar ytterst sällan gammal frånvaro, bara om systemet gjort fel. Tycker att gruppindelningen är onödig, räcker med "lektioner som du inte satt frånvaro på". Gillar att lektionerna man inte satt frånvaro på listas, -det blir psykologiskt</p>

tillfredställande att klicka bort lektioner och hålla listan kort.
4. Hittar direkt till frånvaro i vänstermenyn. Hittar lektionen utan problem i "lektioner du inte satt frånvaro på". Han läser först senare rubriken. Ämnena är inte indelade A, B, C. Han skulle hellre se t.ex. Svenska med 8:orna, Engelska med 6:orna. Hittar även sorteringsmöjligheten.

How comfortable were you with the link names?

1. Länknamnen var enkla att förstå. förstod inte riktigt "grupper du kan sätta frånvaro på". tyckte att det var lätt att hitta. han förstod hela tiden var han var, tryckte dock inte på "grupper du kan sätta frånvaro på" förrän mot slutet. Han undersökte hur man bekräftar en lektionsrapport. gick in i setAttendance och klickade spara. Förväntade sig att flyttas tillbaka till attendance-groups-vyn.
2. Har inga problem med att navigera. Testar att klicka på alla länkar för att orientera sig och lära känna systemet.
3. hade inga problem med länknamnen förutom att. hon gick aldrig in under ämnesgrupper i frånvarovyn. såg aningen förvirrad ut när hon skulle stega tillbaka frånfrånvarovyn till vyn med lektioner.
4. Tycker att det vore enklare om det stod dag i veckan som lektionen är schemalagd i påminnelsevyn.
5. Han har inga problem att hitta runt. tycker att han hela tiden visste var han befann sig på sidan. Då han har ganska god datorvana vet han var knapparna förväntas sitta. Han använder breadcrumbsen för att se på vilket djup han befinner sig. Skulle vilja ha fler ikoner och fler färger för att inte behöva läsa rubriker och utseende.

How did you perceive the *set attendance view*?

1. Han tycker att det verkar mycket enklare än deras nuvarande system. Effektivare. Inte lika mkt. upprepning av data. Efterfrågar ett sätt att ge kommentarer till elever och ange hur lång frånvaron är.
2. Tycker den funkar bra. "Snabb och effektiv". Saknas kommentarfält och giltig/ogiltig frånvaro.
Vill kunna se i rapportvyn:
Vilka var borta den 3:e mars.
Hur mycket har t.ex. Andreas varit borta sen den 1:a januari.
Alla som har varit borta mycket under en tid. (en ranking?)
3. lätt att bläddra mellan elever (pagination). vill ha fritextfält och ogiltig/giltig frånvaro. vill ha länk till en elevs frånvaro/info via namnlänk. Använder ofta kommentarsfältet för att skriva en kommentar (t.ex. bosse störde de andra eleverna under lektionen och jobbade dåligt) till sig själv men även föräldrarna som ser det i veckorapporten. Vill att systemet ska räkna ut hur lång en lektion är, ibland vet man inte då man undervisar för någon annan eller så hinner man inte räkna.
4. Hon har en vana vid att använda tabellistor så hon hittar lätt. Undrar varför pagination ligger däruppe och är på engelska. Hon menar att man sätter antal minuter en elev är frånvarande, inte bara frånvarande/ej frånvarande. Vill

även ha ett bra sätt att skicka frånvaron till föräldrar.

User test 2

Questionnaire

1. Sidan med rubriken "Frånvaro":

- * Är det något fält i tabellen du tycker saknas eller bör tas bort?
- * Finns det något fält i tabellen du tycker borde byta namn eller sätt det presenteras på?
- * Finns det någon ytterligare information/funktion som du tycker kan läggas till på sidan?
- * Är det lätt att förstå vilka lektionstillfällen som visas/inte visas i tabellen?
- * Är det självklart när/varför ett lektionstillfälle tas bort ur tabellen och vart det tar vägen då?
- * Övriga kommentarer:

2. Sidan med rubriken "Registrera frånvaro":

- * Är det något fält i tabellen du tycker saknas eller bör tas bort? Förklara varför!
- * Tycker du att slider-knappen för att registrera "Tid borta" fungerar för ändamålet? föreslå gärna ändringar!
- * Tycker du att textfältet för kommentarer fungerar för ändamålet? föreslå gärna ändringar!
- * Hur tycker du att den gula informationsrutan fungerar? Hur mycket tittar du på om informationen har sparats i den gula rutan?
- * Är det lätt att förstå när dina ändringar på sidan är sparade?
- * Finns det någon ytterligare information/funktion som du tycker kan läggas till på sidan?
- * Övriga kommentarer:

3. Tidigare hanterade lektioner:

- * Är det något fält i tabellen du tycker saknas eller bör tas bort?
- * Finns det något fält i tabellen du tycker borde byta namn eller sätt det presenteras på?
- * Finns det någon ytterligare information/funktion som du tycker kan läggas till på sidan?
- * Är sättet att hitta ett lektionstillfälle tillfredsställande eller behövs ett mer avancerat sökverktyg?
- * Övriga kommentarer:

4. Plats för övriga kommentarer om Doctum:

Answers

1. Sidan med rubriken "Frånvaro":
* Finns det något fält i tabellen du tycker borde byta namn eller sätt det presenteras på? Förklara varför!
-Våra lektioner är oftast 90 minuter långa, men det går bara till 75 min. -Rörigt med lektioner som inte ligger i kronologisk ordning. det blir väl lättare närman har avverkat högen och bara har några färska att sortera, men om man ligger efter skulle det underlätta ifall allt låg kronologiskt -Lektionerna borde ha veckodag och ämne utsatt så att jag vet om jag ex. har Svenska med 5:orna på onsdag. -Bättre namn på lektionerna. Skriv SO helklass. inte samhällsorienterande ämnen
* Finns det någon ytterligare information/funktion som du tycker kan läggas till på sidan?
-Veckonummer till lektionerna!
* Är det lätt att förstå vilka lektionstillfällen som visas/inte visas i tabellen?
-Ja, lektioner som inte satts frånvaro på! -Nej, jag förstår inte vad som är en lektion. Ska jag trycka på ämnet för att komma till lektionen? -Nej, eftersom du satt ut tiderna.
* Är det självklart när/varför ett lektionstillfälle tas bort ur tabellen och vart det tar vägen då?
-Ja, det läggs väl i historiken (?) Men vad hander om man klickar på "lektionen inställd?"
* Är det något fält i tabellen du tycker saknas eller bör tas bort?
Alla nej.
* Övriga kommentarer:
-Har en lektion som börjar nu kl 9.00. Den lektionen dyker uppenbarligen inte upp förrän den är slut och då kan jag inte föra in sen ankomst medan lektionen pågår. Kunde varit användbart. -Bra att lektion som är avklarad försvinner från listan. Då känner man att man betar av skiten. Och fanatstiskt att kunna göra jobbet hemma!!!
2. Sidan med rubriken "Registrera frånvaro":
* Tycker du att slider-knappen för att registrera "Tid borta" fungerar för ändamålet? föreslå gärna ändringar!

-Bra att systemet har koll på hur lång lektionen är. Hel frånvaro är full skrollning - man behöver inte hålla reda på det själv.
* Tycker du att textfältet för kommentarer fungerar för ändamålet? föreslå gärna ändringar!
-Kommentarrutan har begränsat med text. Kan den inte flöda vidare?
* Hur tycker du att den gula informationsrutan fungerar? Hur mycket tittar du på om informationen har sparats i den gula rutan?
-Det är bra att det sparas automatiskt och man får ett "kvitto" på det i och med rutan. Jag var lite skeptisk först men det funkar ju! Många hade inte sett rutan.
* Är det lätt att förstå när dina ändringar på sidan är sparade?
-Bra med en ruta som säger att "dina ändringar sparades". Men det skulle också vara skönt med en "klar-knapp" som man kan trycka till på för att bekräfta att det man ser på skärmen stämmer och ska sparas. Vi är nog lite skadade av Skolreda, då man aldrig riktigt kände att man kunde lita på att en ändring sparats, och bara sparats en gång.
* Övriga kommentarer:
-Dåligt med bokstavsordning efter efternamn. Vi har alla listor efter förnamn och då blir det här rörigt. -Viktigt att eleverna står i ordning efter förnamn (eller att man kan välja) och att man ser hela listan i en spalt. -Skulle helst ha hela klassen på en sida och scrolla ner istället för att byta sida. Både på sidan för frånvaro och frånvarorapport.
3. Tidigare hanterade lektioner:
* Finns det någon ytterligare information/funktion som du tycker kan läggas till på sidan?
-Det behöver bara göras klart så tror jag att det funkar. Intuitivt och effektivt!
* Är sättet att hitta ett lektionstillfälle tillfredsställande eller behövs ett mer avancerat sökverktyg?
-Jag kom inte åt klass 8 för frånvarorapport. Tycker alla undervisande lärare ska komma åt sina klasser, inte bara klasslärare.

Appendix B,

System Screenshots of the second user test



Figure 1, Home view



Figure 2, Attendance reminder view

Doctum (Testversion 1.0)

Registrera frånvaro för Engelska C

Vecka 22, Tisdag, kl 20:14.

- [Hem](#)
- [Frånvaro](#)
- [Ämnen](#)

Elevnamn	Tid borta	Giltig	Kommentar
Agneta Ranerup	0 min ▼	<input type="checkbox"/>	<input type="text"/>
Amanda BjÄrk	0 min ▼	<input type="checkbox"/>	<input type="text"/>
AndrÄ Å...berg	0 min ▼	<input type="checkbox"/>	<input type="text"/>
Andy Aldrian	0 min ▼	<input type="checkbox"/>	<input type="text"/>
Anna Rosengren	0 min ▼	<input type="checkbox"/>	<input type="text"/>
Ann-Helen Antonsson	0 min ▼	<input type="checkbox"/>	<input type="text"/>
Anton Sjästrand	0 min ▼	<input type="checkbox"/>	<input type="text"/>
Barbara Czarniawska	0 min ▼	<input type="checkbox"/>	<input type="text"/>
Barbro Nilsson	0 min ▼	<input type="checkbox"/>	<input type="text"/>
Boel EnbÄygen	0 min ▼	<input type="checkbox"/>	<input type="text"/>

[Första/Föregående] 1, 2, 3, 4, 5, 6, 7, 8 [Nästa/Sista]

Figure 3, Register attendance view

Doctum (Testversion 1.0)

Dina ämnen

- [Hem](#)
- [Frånvaro](#)
- [Ämnen](#)

Ämne	Årskurs
Engelska C	3
Svenska A	1

Doctum - Testversion 1.0

Figure 4, Your subjects view

Doctum (Testversion 1.0)

Frånvaro - Tidigare hanterade lektioner

[<< Tillbaka till Frånvaro](#)

- [Hem](#)
- [Frånvaro](#)
- [Ämnen](#)

Välj ett ämne

Ämne	Årskurs
Engelska C	3
Svenska A	1

1

Doctum - Testversion 1.0

Figure 5, Earlier handled lessons view

Doctum (Testversion 1.0)

Frånvarorapport

[<< Tillbaka till Frånvaro](#)

- [Hem](#)
- [Frånvaro](#)
- [Ämnen](#)

Välj elev

Elevnamn	Personnummer
Adam Boklund	198205281001
Alan Carlson	198205281004
Anders Wallin	198205281009
Andreas Larsson	198205281021
Anna Nordberg	198205281024
Anne Stillbäck	198205281027
Annika Svensson	198205281030
Ariuna Geir Aasehaug	198205281033
Barbro Neiberg	198205281036
Björn Byström	198205281039

[Första/Föregående] 1, 2, 3, 4, 5, 6, 7, 8 [Nästa/Sista]

Figure 6, Attendance report view 1

Frånvarorapport för Adam Boklund

[<< Tillbaka till välj elev](#)

- [Hem](#)
- [Frånvaro](#)
- [Ämnen](#)

Lista frånvaro mellan två datum

October 2009							November 2009						
Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa
27	28	29	30	1	2	3	1	2	3	4	5	6	7
4	5	6	7	8	9	10	8	9	10	11	12	13	14
11	12	13	14	15	16	17	15	16	17	18	19	20	21
18	19	20	21	22	23	24	22	23	24	25	26	27	28
25	26	27	28	29	30	31	29	30	1	2	3	4	5
1	2	3	4	5	6	7	6	7	8	9	10	11	12

Visa rapport

Doctum - Testversion 1.0

Figure 7, Attendance report view 2

Frånvarorapport för Adam Boklund

mellan 2009-10-15 och 2009-11-22

[<< Tillbaka till välj elev](#)

- [Hem](#)
- [Frånvaro](#)
- [Ämnen](#)

Frånvaro (min)	Total lektionslängd	snittfrånvaro (min)	frånvaroprocent (%)
0	0	0.0000	0.0000

1

Doctum - Testversion 1.0

Figure 8, Attendance report view 3