Browser based room booking system

Project:	Bachelor project
University:	ITU 2011
Advisor:	Peter Sestoft

Kristian Klarskov Marquardsen	kkma@itu.dk	220387-1611
Frederik A. Wordenskjold Noerregaard	fawn@itu.dk	140987-1727

Contents

Chapter 1

Background

1.1 Introduction

In this project we have designed a user-friendly room booking experience for the IT University[1], and provided the concept for a solution to the complex task of assigning rooms to courses. The current process of assigning rooms to both students as well as courses, is both tedious, inefficient and without proper support. At the moment, it is hard to get an overview of how rooms are used and how efficient the current allocation is. This makes it hard for the staff to exploit the full capacity of the university, which is of high importance to expecially ITU, because of the rather limited number of rooms and auditoriums.

We want to construct a program that centralizes this process, and aids the persons responsible as much as possible. Such a system would make the day to day task of booking a room for group work, seen from the perspective of the students, much more flexible. The harder to grasp tasks such as completing the room assignments for a whole semester should be supported by the system, to make it easier for the employees to make the right decisions. As it should be possible to book a room at home, on the way to school or on site, we have implemented the system as a web-based solution.

[1] IT University will henceforth be refered to as 'ITU'

1.2 Current situation

As of writing, room management at the ITU is split between the Study Administration and the Facilities Management. Within those departments there is another division of labour which all in all adds to the confusion of handling the limited rooms efficiently.

1.2.1 Facilities Management

The information desk, controlled by FM, is currently in charge of the day-to-day use of the meeting rooms. When a student or teacher wishes to reserve or use a room, the FM employee will refer to a binder with a paper-overview of current status, and if they succesfully find an available room a form is filled out, photocopied and archived.

Aside from this, another part of FM is in charge of assigning offices and workspaces to teachers, teaching assistants and researchers. This is done through a locally stored Microsoft Access database, and is handled primarily by one person.

1.2.2 Study Administration

In regards to room management, the Study Administrations biggest challenge is the planning of a semester. First step is collect data from the course database to see which courses will be held in the coming semester. When a list of courses have been compiled, a system called Tabulex[2] is used to create a draft schedule so no overlaps of teachers and courses required by several tracks[3]. The output from Tabulex is finally used to manually create the room assignments for a standard week of the semester.

The other function of the Study Administration is to assign rooms required by exams, for both the examnation itself and for the prepation-time required by some exams. Unfortunately, not all courses have ended by the time exams begin, so apart from rooms being occupied the problem of privacy and noise presents itself.

TODO:**Perhaps reformatting and elobaration**

[2]http://www.tabulex.dk/

[3]'Track' is the term used to describe a semesters worth of courses in a specific direction, eg. the 'Software Engineering' direction of the 'Software Development and Technology' MSc education

1.3 Scope

Our approach to designing this system has been primarily focused on usability and all that entails. Tedious and de-centralized management of a single resource is both time consuming and highly error-prone. To exemplify a solid solution to this problem, we have narrowed our vision to the most used area (day-to-day booking) and the most tedious area (semester planning).

The simple booking have been succesfully prototyped, whereas the semester planning have only been designed and developed theoretically, and not implemented. Doing the full scale implementation of the semester planning would of course be a possibility, but it is not crucial to prove our point of data presentation and usability and we have therefore chosen to perform all the analytical work on it instead.

As mentioned in the original problem definition, we have also elected not to include an administration panel for super-users, as it would serve little purpose without all the features of the full system.

TODO:**Limitations/inclusions of our project work goes here**

Chapter 2

Analysis

When unifying the functions of several departments and optimizing the overall process at the same time, a multitude of non-triviel problems will need answering. Through this chapter we will go through the discussions made for and against our choices.

2.1 Scope

Our approach to designing this system has been primarily focused on usability and all that entails. As described in 1.1, the current situation holds multiple challenges. Tedious and de-centralized management of a single resource is both time consuming and highly error-prone. To exemplify a solid solution to this problem, we have narrowed our vision to the most used area (day-to-day booking) and the most tedious area (semester planning).

The simple booking have been succesfully prototyped, whereas the semester planning have only been designed and developed theoretically, and not implemented. Doing the full scale implementation of the semester planning would of course be a possibility, but it is not crucial to prove our point of data presentation and usability and we have therefore chosen to perform all the analytical work on it, and leave out the actual implementation.

As mentioned in the original problem definition, we have also elected not to include an administration panel for super-users, as it would serve little purpose without all the features of the full system.

2.2 Requirements

Since the ITU does not currently have a full system, we have put up initial requirements for such a system. The requirements listed is the functionality that, when combined, provide a solution to the problem at hand.

2.2.1 Use cases

The booking of a room have four parameters; the week of the year, the day of the week, the time of day and of course, the room. With these four in mind, the searching, browsing and organising of rooms pose some relevant questions.

Legend	
W	Week
D	Day
T	Time
R	Room
Main case	High occurance
Edge case	Rare occurance

With one parameter locked

Main cases:
W - "I need a room sometime next week"

Edge cases:
D - "I need a room on a wednesday sometime"
T - "I need a room at 12 o'clock sometime"
R - "I need Aud. 4 at some point"

With two parameters locked

Main cases:
WD - "I need a room at thursday next week"
WT - "I need a room at 9 o'clock next week"
WR - "I need Aud. 1 in 2 weeks"
DT - "I need a room mondays at 8 o'clock" (W free, so meaning a span of weeks needed here)

Edge cases:
TR - "When is Aud. 4 available at 9 o'clock?"

With three parameters locked

Main cases:
WDT - "I need a room at 12 o'clock on friday next week" (Expecting this to be the most used search of them all)
WDR - "I need room 4a14 this friday"

Edge cases:
WTR - "I need Aud. 3 at 9 o'clock next week sometime"
DTR - "I need room 2a09 on thurdays 11 o'clock" (Probably to be used along with a semi-locked W (span))

2.3 Design decisions

2.3.1 Platform

Choosing a webbased solution over a desktop application was both a choice of interest and of usability. Accessibility is a crucial factor in usability, and by allowing end-users to access the system instead of having to go through the bottleneck of a single department, the workload is both distributed and handled immediately, as opposed to being dependant on an employee from said department to perform the whole process associated with locating and booking a suitable and available room.
The drawbacks to TODO:**finish this sentence**

TODO:**How to implement, discussion of for and against our choices of platform, language, etc etc. Possibly split this in to several sections**

2.3.2 Concurrent room selection

The handling of concurrent users is always an issue in webbased services, and more specifically in our case it presents itself when multiple users wants to book a room at the same time. No solutions are perfect, and we've tried discovering the pros and cons of the solutions we considered.

Possible solutions

These solutions are based on the the case where users **X** and **Y** are searching for rooms at the same time. They could obviously be extrapolated to cases where even more users are interacting.

*Make all rooms that fits **X**, unavaliable for **Y** while **X** is deciding on the which room to use.*

- Cons:
  - If all rooms fit the search of **X**, no rooms will be available for **Y**
- Pros:
  - Easy to implement.
  - Extremely unlikely to encounter dead ends.

*The screen which shows avaliable rooms, of user **Y** should update when user **X** makes a booking, to make sure **Y** doesnt hit a dead end.*

- Cons:
  - Increased number of requests to the server.
  - Dead ends possible.
  - Ultimately, it is a tradeoff between probability of dead ends vs. server request-rate.
- Pros:
  - High usability and responsiveness, resulting in a pleasant experience for the user if no dead ends are hit.
  - More flexible; allows several users to have many choices at once.

Chapter 3

User interface

TODO:**general introduction to this section, and why it warrents a whole chapter**

3.1 Challenges

TODO:**What challenges must our user interface overcome?**

3.2 Usability

TODO:**Usability discussion and documentation for findings from our tests**

3.2.1 Test af mockup1

Tasks to perform
1 - simple: "Book a room today"
2 - simple: "Book a room tomorrow"
3 - special case: room not available current day: "Book room x for use as soon as possible"
- ...

Questions to ask
- "cancel" or "back" ? : "what do you think this button does?"
- Calendar - is it obvious?

Lead-in
1. You and your group have met, and need to find a room to conduct group-work.

2. Your group has finished work today, and decided to reconvene tomorrow. You have been assigned to arrange for a place to work tomorrow.

3. Your group will be having it's exam in auditorium 3, and you wish to prepare for it in the same setting as the actual exam.

11 - 11.20: Jakob 27 DMD 2. semester

1. Find a room
Trykker på Find me a Room siger trykker vel her. Vælger tid fra og til.

2. Rum i morgen
Trykker på find me a specific room. Finder hurtigt ud af at han nok skulle have valgt vha. kalenderen.

3. Eksamen i auditorie3 en anden dag.
Trykker på auditorie3 på kortet. Trykker på another day.

4. Find rum næste fredag.
Find me a specific room. Vælger "projektor", og er nice. Spørger om man skal vælge tid her.

5. Find et lokale til mere end 50 mennesker på en onsdag.
Find me a room, cancel, find me a specific room. Task failure. Går tilbage til kortet. Trykke på et af auditorierne. Siger at han nok i stedet ville trykke på kalenderen.

6. Komsammen for førsteårsstuderende.
Trykker på aud1 på kortet. Bruger kalenderen til at vælge datoer, for at se ledigheden.

Jon, 23, SWU 6. semester

1. Find me a room, OK.

2. Bruger kalendervælgeren til at finde datoenm find me a room, OK.

3. Vælger dato gennem kalenderen, find me a specific room, OK.

4. Som 3.

5. Find me a room. Cancel. Trykker på w på kalenderen. Task failure, specific room. Vælger "auditorie" og ikke capacity. OK.

6. Trykker på aud1 på kortet. Trykker på datoen, trykker på start-tid og slut-tid. Trykker på book.

3.3 Design process

To our project, the design of the graphical user interface is one of the most important things. Instead of focusing on hiding the difficulty of the problem by obscuring it with automization and huge calculations, we seek to present suggestions and options for solutions in such a manner that a human being can lay the final touch. It has become apparant through our tests and interviews that no solution granted by a computer alone will be sufficient - there will always be conflicts needing a (to a computer) not obvious resolution, and even in the simplest of cases where no conflicts occur, human polish and/or acceptance is key.

We've undergone many iterations of each and every screen in the system, both those implemented in the prototype and those only on mockup stage. What follows is a tour through the process of the design.

3.3.1 Trimming the fat

The design of a page begins with deciding what function it should serve in our system, eg. a page to book a room for 2 hours. With that constantly in mind, we attempt to find all the possible things that particular function COULD include. Some of them are extremely basic and obvious, eg. it needs to display which room you are booking, others are more complicated, as for example the ability to check whether this room is available on a regular basis.

Through the process of 'overthinking' each page, we came up with smarter solutions to cover several different items on the could-list, or found out we needed a new page to handle all the "advanced" settings as an example. Once we had settled on the functionality of the page, we began piecing the puzzle together.

3.3.2 Room booking design

For a full history of our mockups, please refer to

TODO: **indsæt billede af første mockup til book rum**

As seen above, the lower right panel was thought to be in charge of all interaction, trying to give the user the impression that they never left home, and to make sure no-one got lost. Our first usability tests quickly revealed that we had been mistaken. All the test subjects felt overwhelmed by the amount of options, and due to all the clutter did not spot the vital functions.

TODO: **indsæt billede af næste mockup til book rum & billede af weekoverview mockup**

This time around, we tried making it as basic as possible and simply presenting the user with a dialog (not a pop-up, just an overlay) for selecting how many hours from now the room should be booked. The flaw with this was that noone was able to figure out the availability of a room in the near future. This is when we realized that the best successes we've had with room booking was the first draft of the week-overview screen. All the test subjects used the 'matrix' overview both creatively and effectively, so we decided to try and utilize that success to the maximum by redirecting even a simple booking task to the same screen. The only exception we have made to this, is when using the "book now"-functionality, which will simply book a room for 4 hours and notify the user which one they have been assigned.

TODO: **indsæt Billede af færdig book skærm**

The final design to book a room, as shown above, grants availability overview of 7 days, starting with current, and supports both dragging of the mouse to select a timespan, and clicks[1] as we had experienced both during the course of our tests. To assist the user in figuring out how it works, we naturally have highlighting of both the day and time when mouse is drawn across the matrix.

Søren Lauesens —- first law of usability states that a heuristic evaluation only has a 50

3.3.3 Layout

Being a website poses some extra considerations - namely those of browsing habits. Two examples we will refer to, and assume basic familiarity with, in this section are wikipedia.org and google.com TODO: **indsæt små billeder af wikipedia og google med F-pattern eyetrack**

[1] Click for start time, click again on end time

3.4 Proposed design

TODO:**Explain and illustrate the proposed design**

Chapter 4

Implementation

4.1 Datastructures

TODO:**Introduction to this section**

4.1.1 Datamodel

TODO:**Explain our database design and datamodel**

4.1.2 Algorithms

TODO:**Outline the important algorithms in our program, and why they are awesome**

4.2 Server-Client relationship

TODO:**Explain the design; what we do, and what lift handles for us.**

4.3 Lift

TODO:**This section probably doesn't belong here, but it needs to be somewhere.**

Chapter 5

Evaluation

5.1 Perspective

TODO:**Take a step back and look at our work as a part to a whole. Tabulex, FM/Study admin etc.**

5.2 Possible improvements

TODO:**If this was a commercial product, what would we change/add/fix etc.**

5.3 Expansions

TODO:**Consider if this section is needed. Perhaps it could fit into the above. If not, we could talk about ridding the need for tabulex, centralizing everything in our system and so forth.**

5.4 Reflection

TODO:**Reflect on the process.. hurp derp**

Chapter 6

Conclusion

TODO:***drumroll* Round off the paper in an elegant, humble, and yet awesomely awesome way.**