CHALMERS

Investigation and development of user design utilities for widget based web services

Master of Science Thesis in the Programme Computer Science and Engineering

SAMUEL SVENSSON

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Göteborg, Sweden, May 2009

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.

Investigation and development of user design utilities for widget based web services

Samuel Svensson

© Samuel Svensson, May 2009.

Examiner: Bror Bjerner

Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden May 2009

Abstract

Amadeus e-Travel Management - AeTM - is a web-based booking application for personal transportation and hotels. The way the application is working have not changed much since 2003, but since then a lot have happened with web applications in general and user interactivity in particular. This thesis presents how user interactivity with web 2.0 techniques can be used in the AeTM application, in order to allow the users define the layout and design of the application.

This thesis presents a number of solutions, compares them and also a successfull implementation of the best solutions of the application. The AeTM User Experience Team thought the solutions were interesting and with some minor changes ready for further development and possibly future implementation on the AeTM service.

Preface

This dissertation in Computer Engineering at the department of Computer Science and Engineering at Chalmers University of Technology was done during the spring and summer of 2008. The thesis is a result of an internship at Amadeus' site in Sophia Antipolis in France in the Amadeus e-Travel Management User Experience team.

Examiner for the thesis was Bror Bjerner, lecturer at the institution of Computer Science and Engineering at Chalmers University of Technology. During the first half of the internship the supervisor for the internship was Patrick Chalony, the second half was supervised by Nicolas Bijaoui.

Samuel Svensson, Göteborg 2009-01-10

Contents

1 Introduction

The purpose of the work of this thesis is to research and develop extensions for Amadeus' e-Travel Management web application, from here on referred to as AeTM. The goal of these extensions, is to give the user a possibility to configure the presentation of content, with focus on giving the user a powerful and easily used tool, with as small requirements on the user's knowledge of web design as possible. The major challenge of the thesis is, to create an intuitive and user friendly environment and also find the balance between ease of use and high detail configurability. Other problems are, to analyse the AeTM-web application and determine how the user interfaces are best integrated with the existing product. Also to some extent, how the AeTM-web application will need to be changed to be mergable with the suggested extensions. Information about the company, the product, and a more detailed description of the extentions is presented later in this chapter.

Structure of the report This report will first introduce the company and the product. Chapter two includes analysis of the user, the interface and requirements. In chapter three, the methods and processes used during the project, will be described. Chapter four presents the solution suggestions and further analysis. Chapter five results, followed by discussion of the results in chapter six. The final chapter, number seven, presents conclusions and recommendations for future development.

1.1 Background

The possibilities of what can be done with web applications have evolved over the years. The initial web sites were static documents, where the user interaction were limited to clicking on hyperlinks to access other documents. Today the static documents still exists, and still serve their purpose, but web services also have the potential to be much more complex. A lot of applications that we are used to run locally, are today available as web applications. Examples of these kinds of applications are Google docs[1] or SUMO Paint[2].

1.2 Amadeus

Amadeus is mainly a Global Distribution System - GDS - to book and sell tickets for multiple airlines and travel agencies. Today over 500 individual airline companies and over 90 000 traveling agencies are using Amadeus' services. It was founded in 1987 by an alliance between Air France, Lufthansa, Iberia Airlines and Scandinavian Airlines System.

[1]Google docs is a Web based word processing and spreadsheet service, that has been available since early 2007.[7]

[2]SUMO Paint is a Web based image editor. The editor was released as an open beta in June 2008.[9]

The company is also involved in other areas, but almost always with the GDS as the foundation. One of Amadeus departments is SEP - Sales and e-commerce platform - which provides booking systems for airlines and traveling agencies. SEP is primary working on two products, the AeTM and the Amadeus e-Retail Engine. The owner of Amadeus is Amadeus IT Group, with their headquarters in Madrid, Spain.

1.3 The Amadeus e-Travel Management

The product is a web-based solution for companies, which have employees that need to travel in their work. The company books the trips for the employees or lets the employees book their trips themselves. Thanks to the application, the company using the system, is able to regulate, what trips the employees are allowed to book. Limitations to certain airline companies or other rules can be applied.

The current product is based on BEA WebLogic Portal application server[3] and have been developed on that platform since 2003. The presentation layer is made of a wide range of JavaServer Pages (more than 1,500 files) as well as a set of Javascript files (approximately 25,000 lines). BEA WebLogic Portal uses some concepts which the web interfaces are based on. The main subjects of this thesis are portals and portlets. A portal is a major framework, which in its turn handles various portlets. The portlets are more or less complex programs, similar to the today more well known web-widgets or gadgets, but the portlets can also have a more complex functionality compared to a normal widget.

1.4 The task

The goal of the project is to study and propose some alternatives and enhancements to the current design of the AeTM Web product. The overall task is, to create a solution for easy customization through the browser itself. For some configuration aspects, the user needs not to have required knowledge about web designing or coding, while some more high detail configuration aspects could require the user to have some basic experience of web interfaces. With the vast quantity of elements that should be configurable through this interface, the challenge is how to make a clear and intuitive interface, without compromising the configurability.
The initial rough task description:

> **Portlet display and positioning** AeTM is based on portal technology and each logical module of one given page is displayed inside what we call a portlet. The portlet positioning is currently predefined and the same goes for all users. The goal of this study is, to enable the user to decide dynamically the positioning of one given portlet, i.e. select and move a portlet with the mouse or place the portlet

[3] BEA WebLogic Portal is a framework for creating, deploying, and managing multiple enterprise portals. It is owned by the Oracle Corporation.

at some authorized locations predefined in the layout. If possible, the study can include the possibility of persistency.

Layout customization To give the user, the possibility to define his/her own layout for one given page. This study will include a list of proposals on what is generally expected for layout customization and some feasibility studies.

Look-and-feel customization To give the user, the possibility to customize the look-and-feel of some identified parts of one page. A page is composed of a header, a footer, a navigation bar, and the central part, composed of many portlets. The goal is, to study the possibility to upload some html/css code to customize the header/footer navigation bar and apply alternative graphical representations of some portlets.

For the original rough task description see appendix C. The two last tasks were removed at an early stage. This was due the fact that working with all five tasks would not be feasible in with the within the time span of the project.

1.5 Limitations

There are some limitations of the types of techniques that can be used. The most common browsers of the biggest operating systems shall have A-grade browser support[4]. The application should also not need any separately installed software, such as for instance the Adobe Flash Player. Since the customization extension is supposed to run together with the the AeTM-web application, it should preferably be implemented in the same program or script language as the existing product.

Since there is a vast set of good open source solutions of increasing the ease of creating web applications and improving user interface functionality, these tools will be used to the extent proven useful, e.g. user interface libraries. Since these problems are of no interest of the project at hand, no work will be done creating these kind of tools, unless no suitable opensource solutions are found.

[4]To have A-grade browser support means to provide identical experiences for the user, independently of the chosen browser software.

2 Analyse

2.1 The user

The applications to be made, will not be available to any user, only to users with an administrative role in the community. Since it is up to the company using the AeTM product, to assign someone with this role, this does not necessarily mean that the administrator is guaranteed to have any specific knowledge or training.

No focus will be put on having the product adjusted to fit any user disabilities, such as any visual handicaps or any other physical disabilities. The user is expected to use a stationary computer or laptop without any problem.

Prerequirements To the extent it is possible, the user is not to be required to have any previous experiences or knowledge of webdesign. The user is expected to have a basic computer experience from normal usage of any operating system and any web browser. In order to use some more advanced functions of the application, the user may be required to have a more technical knowledge; the requirement may be to understand web applications and the structure of HTML-documents and CSS. Since it severely reduces the number of users that can operate the application, this kind of prerequirement is to be avoided as much as possible.

2.2 The AeTM user interface

The AeTM-web application is a system based on BEA WebLogic Portal. The portal is a framework, that provides content and functionality through a web interface. The areas of the user interface can practically be divided into four parts: header, footer, navigation bar, and portlets.

Header: On the top of the site there will always be a header present. The header is related to the community, which is using the service. Some of the headers' appearance can today easily be modified by the community administrator, by loading background images. Also, the header holds other content of a more static nature.

Footer: On the bottom of the site there will always be a footer visible. The footer is more or less of a completely static nature, which the administrator of the community have no possibility to affect. It holds information in text format, related to policy and copyright of the service.

Navigation bar: The navigation bar is split into two types, main navigation bar and sub navigation bar. The main navigation bar provides links to access different categories of functionality. The sub navigation bar is used to access different specific functions within the current category. The accessed functionality is provided on different pages through portlets.

Portlet: The portlets hold the content of the site and most of the sites functionality. There are no limits of the content or functionality provided by the portlet. Portlets are placed in the area between the header and the footer in a table structure, where cells have a defined width.

2.3 What functionality will the system provide

Portlet display and positioning: Since the current solution has a completely static structure, that can not be changed by the user, the goal of this part is to allow the user to declare the portlets to be displayed and how they are positioned.

Layout customization: Since this part is closely connected to the Portlet display and positioning, it defines the layout, where the portlets are to be placed. The goal is to make it possible for the user to create a layout of positioning.

Look-and-feel customization: To make it possible to design changes to some identified parts, all of them mentioned and described in 2.2 The AeTM user interface. The user is to be able to freely customise the identified area with all the parameters available in HTML and CSS. This might require the user to have some knowledge of these areas.

2.4 Usability requirements

Intuition and configurability: The user should never hesitate on, what to do when presented with the application. The interface that is to be created, shall feel natural to the user, accordingly to the purpose of the specific tool and how the tool is to be used. An action performed by an user, should have a corresponding response. The user shall always be aware of what is happening and never hesitate about, what the applications is doing with the provided instruction.

Response time: The response time of all actions are preferably instant when possible. Thus, as much work as possible will be done locally on the client machine. Client-server communication will be necessary for the applications, but is to be avoided if operations can be done locally instead. This is to lower the response time of the applications. When the client initiates communication with the server, while it waits for the server response, the clients applications should not be locked. It should also be clear to

the user when a transfer is initiated, so there is no hesitation whether the application might have crashed, that the request from the user was done in the wrong way, or was not registered.

Integration: The existing product and the configuration extension is to be integrated in a good way. The tools made, shall be a natural part of the AeTM-web product and the user shall experience them as one unit. Preferably the application should not only be well merged in the user aspect, but also a proper extension in a coding aspect.

2.5 Suitable tools

Based on the requirements, see 1.5, the application is not to rely on some proprietary multimedia application player such as Adobe Flash Player, or similar extensions.

Since the application is to be merged with the existing product, it should be written in the same programming or scripting language as the existing product. The AeTM-web application is today mainly based on JavaServer Pages, Java Scripts, and ordinary Java.

JavaServer Pages: A Java technology to create dynamically generated HTML, XML or other types of documents in response to a Web client request.

JavaScript: A scripting language most commonly running on client side to enhance the functionality of web applications.

Asynchronous JavaScript and XML: More known as Ajax, brings the possibility to have a webservice, that can retrieve data from the server asynchronously in the background, without interfering with the display and behavior of the existing page.

Yahoo! UI Library: An open source solution used for creating rich interactive web applications. It is based on Ajax, DHTML and DOM scripting. The library is having A-grade browser support for all major browsers in their most used operating systems, for detailed information see A-Grade Browser Support[2]. For more detailed information about the Yahoo! UI Library in general see The Yahoo! User Interface Library (YUI)[1]

3 Method description

This part of the thesis describes the methods used in this project. Areas, that will be included, are the planning process, the development models, and the interaction methods, used during the development and implementation of the application.

3.1 The planning process

The three different areas of research were all put in a development cycle. The cycles were executed one after another and only after one cycle was considered finished, the next one in the order was initiated. Each cycle had an internal process as well. For further information about the internal process, see chapter 3.2 Development model.
The three areas to be researched were:

- Portlet display and positioning
- Layout customization
- Look-and-feel customization

The subjects are described in more detail in 2.3.

In the beginning of the internship there was an estimation of how much time the different areas of research, studies, and prototypes would take. This rough schedule was used during the project as a guideline of how the time should be spent and how to know approximately, when a certain phase was expected to begin and end. The schedule was not particularly strict and as some phases took more time than expected, other phases has to be finished quicker. More information is available in appendix B.

3.2 Development model

The development model of this project is divided into two separate iterative cycles, research and prototyping. The research cycle is the first and must be finished in order to procede to the prototyping stage.

3.2.1 Iterative design

During the project an iterative design paradigm was used. The following is a description of the process, from "Agile and Iterative Development: A Managers Guide"[3] by Craig Larman. :

Iterative development is an approach to build software (or anything), in which the overall life cycle is composed of several iterations in sequence. Each iteration is a self-contained mini-project, composed of activities such as requirements analysis, design, programming, and testing. The goal of the end of an iteration is an iteration release, a stable, integrated and tested partially

complete system. To be clear: All the software across all the teams is integrated into a release at each iteration. Most iteration releases are internal, a baseline primarily for the benefit of the development team – they are not released externally. The final iteration release is the complete product, released to the market or clients.

3.2.2 Research

During the research step, one solution or several alternative solutions are found for the problem at hand. The result may either cover the whole problem or most likely during the first iterations, cover a subset of the problem. The research step covers everything from user interaction to implementation problems.

During the review step, all alternative solutions are compared and discussed with the supervisor and/or a part of the AeTM User Experience Team. Compared to each other and with respect to usability and integration with the AeTM product, the alternatives may be eliminated or become subjects of further, more detailed, research. If a part of the problem is considered to be finished and not need any further research, it will be documented and left outside the research loop process.

For a detailed description of the methods used for reviewing usability during the research stage see Paper prototyping (3.3.1).

The research cycle goes on, until there is enough material for a complete design document and the requested functionality of the application is fulfilled.

3.2.3 Prototyping

During the prototyping step the code is written based on the design document. The resulting program is not necessarily completely finished, when it leaves the prototype step of the review and testing, but there will have to be enough relevant modules of the program to perform the tests. The reason for reviewing an unfinished program is, that some aspects of the usability and experience of the interface may be different from the predicted.

During review and testing, the prototype is presented to the supervisor and/or a part of the AeTM User Experience Team. During the review and testing phase, the usability and functionality is tested and discussed. Any part of the application that is not working in a satisfying manner, is analysed. Resulting conclusions are taken into account in the following redesign step.

At some points during the process, when considered suitable as new content was added, a person who has not been in the process of testing previously, is introduced to the application. With a short description of the application's purpose and in some cases a brief description of how it is intended to work, the user may freely use the application. Afterwards, he or she may leave any thoughts on whether the interface was working in a satisfactory way, e.g. if there were some user actions that lead to other results than those the user expected, or any other thoughts on the application.

During the redesign step, the the application is changed based on the results from the review and testing phase. Before the implementation step commences, any changes of the application are written to the design document.

3.3 Testing methods

The methods used to test the interaction, are techniques, that belong to the User experience design and Human-computer interaction disciplines.

3.3.1 Paper prototyping

Paper prototyping is a user-centered design process. The process is useful in helping the developer to create interfaces, that meets the users's expectations and needs. It is a rough and cheap prototyping style, which involve creating drawings or models with pen and paper or other physical objects. The most important areas in paper prototyping are:

Communication in the Team One area is team brainstorming, in order to collect ideas of how the interface might look. Step by step the interface is created meeting the expectations of all team members. Usually use cases[5] are played through, in order to detect possible pitfalls.

Usability Testing The prototypes can be used for usability testing, preferably with five real users.[6] The users perform real tasks by interacting with the paper prototype. Another person is manipulating the prototype in the way the real application is intended to react on the user input. Even though this method seems rather primitive, it is very successful in discovering usability issues early in the design process.

Design Testing The user is presented a high-fidelity design paper mockup. Among other relevant questions, the users are asked to identify the elements that are interactive. This is a very useful test especially for web services, where the user is to determine the main navigation and other clickable elements.

[5] A use case is a description of a systems behaviour as it responds to a request that originates from outside of that system.

[6] According to Jakob Nielsen, the best number of real users for usability tests are five persons. Smaller groups tend to miss some flaws in the design, and bigger groups are not that much more likely to find more design errors.[5]

Information Architecture By testing with paper prototypes, the expected information architecture of a software can also be tested. The users are presented the prototype and asked how they are to obtain certain functionality or information. According to the rate of correct answers, the information architecture is determined to be sufficient or to be redefined.

3.3.2 Expert review

Reviewing the prototype was done with the expert review technique. A technique recommended by Albert N. Badre in his book Shaping Web Usability: Interaction Design in Context[4]. :

Expert reviews can be either comprehensive, covering every relevant usability guideline, or heuristic, limited to a select number of high-level rules (Molich and Nielsen, 1990)[7]*. One or more experts perform comprehensive reviews. The experts evaluate the Web site using detailed design rules and guidelines, such as consistency of layout and the providing of feedback, to guide their judgments. The best reviews are performed by those who are experts in both Web usability design and are also domain experts (Nielsen, 1992)*[8]*. It is therefore recommended that whenever possible, a committee of usability experts and reviewers with subject-matter expertise perform the reviews.*

The experts of the reviews were members of the AeTM User Experience team.

[7]Heuristic evaluation of user interfaces. Proc. ACM CHI'90 (Seattle, WA, 1-5 April), 249-256.

[8]Finding usability problems through heuristic evaluation. Proc. ACM CHI'92 (Monterey, CA, 3-7 May), 373-380.

4 Suggestions of solutions

4.1 General

Since an user works with any application, the best way the user knows the results of the actions, is to provide the user with feedback, that directly corresponds to the result. Preferable is to have a constant or at least a high frequency of feedback, where the feedback is the actual result or a representation as close to it as possible.

Meeting the requirements from 2.4 Usability requirements of intuition and merging, the goal is to let the application run completely simultaneously with the live instance of the AeTM-web application. Editing the AeTM interface seeing the site and the results of all actions live as the user requests are performed, is the most preferable representation and is applied to all solutions except for Look-and-feel customization 4.4, since it require user requested previews.

In this section PRO&CON lists[9] will be used.

4.2 Portlet display and positioning

Since the user is to be able to choose the portlets to be viewed and how they are to be positioned, four alternative solutions were created and finally compared. The problematics around browser behaviour and HTML interpretation are not considered here, but are underlying factors of the described behaviour of the various techniques.

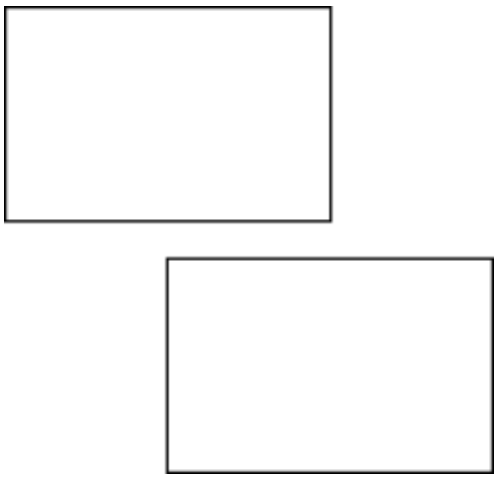

Figure 1: Absolute position: Portlets as placed by user.

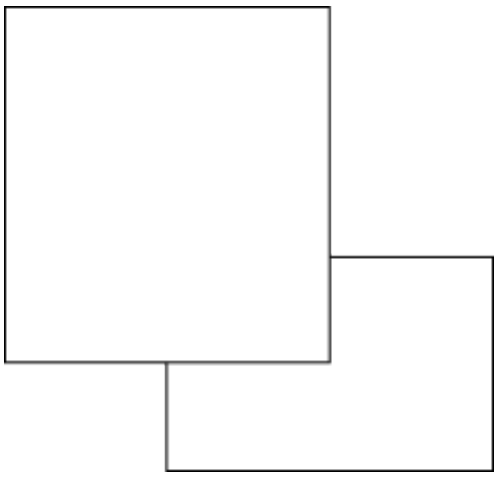

Figure 2: Absolute position: How coverage problem may occure with dynamic content.

[9] List of all the good and bad aspects of a solution. PRO is a benefit and CON is a disadvantage.

4.2.1 Absolute position

One alternative of how portlets are to be placed and arranged, is to let the user specify the exact position of each portlet. Combined with letting the user define the width and the height of all portlets, this approach gives the user a tool, with which any layout can be made. It is possible to arrange the portlets down to the resolution of pixels or by a more rough grid system, if a lower resolution is desired. This interface meets with the desired attributes mentioned in the beginning of this section. An example placement can be seen in Figure 1.

PRO Since the portlets and their behavior are represented and managed in a way that correlates to how windows are arranged and sized in our most commonly used operating systems, an application based on this concept is very intuitive and easy to use. The possibility to drag the borders or the edges of the portlet in order to resize, is a well known concept for most users.

PRO Since each portlet have its unique fix parameters for each page, this solution removes the need of a separate layouting tool.

CON Since the portlets are being placed and sized by the user, the problem is that the content inside the portlet won't always necessarily fit. The content of most portlets are of a dynamical nature, so the amount information to be presented in a portlet may vary. This will result in either that content that won't fit in the portlet area simply not be displayed or that scrollbars are to be applied to each portlet with too much information. The problem with dynamic content leading to portlet intersection can be seen in Figure 2.

CON Preventing content from being shown in a portlet, is not an acceptable solution, since it results in lost information or even lost functionality.

CON Applying scroll-bars to a portlet will not remove information or functionality, but instead result in problems for the user to get a good overview, since some content may be hidden.

CON To allow the portlet to adjust its size to be able to display all of the content, is the superior solution in a functional and user experience perspective, since the other alternatives makes the portlets hard or even impossible to use. The drawback of this solution is, that the resulting layout is hard or even impossible to predict. Since the portlets are placed with a fix position and the sizes have the possibility to dynamically change with the content, there is a risk of portlets expanding into each other. This could result in lost information, which is not acceptable.

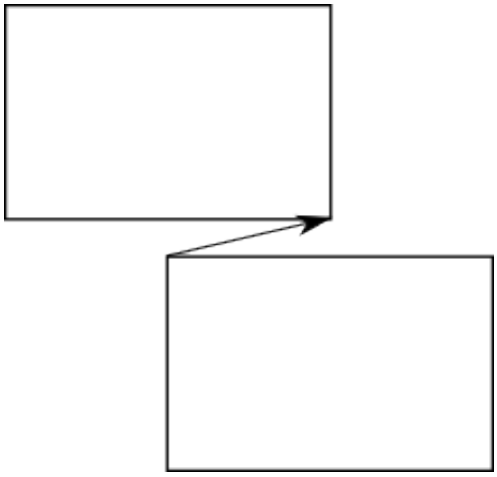

Figure 3: Relative position (manual): Portlets as placed by user and relations declared.

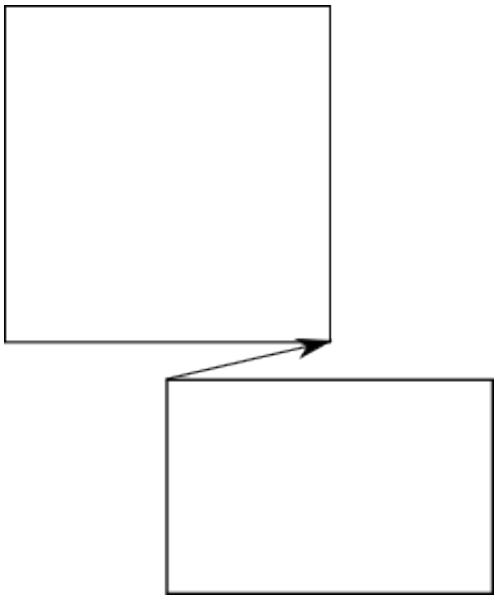

Figure 4: Relative position (manual): When declarations are made correctly the portlets works well with dynamic content.

4.2.2 Relative position to other content (manual declaration)

A second alternative, is to introduce a combination of absolute and relative positioning. All portlets are placed in the same fashion as in absolute positioning. The user specify the exact positions for the portlets, but when placed the portlets are declared to be in relation to a corner of another portlet. Since the portlets are adjusting to each other with respect to the sizes, the problem with overlapping portlets will be avoided. Since there is only to be one point of reference, dynamic expansion of the portlets may only be in on the the dimensions of width or height. The most preferable will be to allow expansion of the portlet height. This is due to the norm of scrolling down and not sideways, while using web applications. An example of placement and declaration can be seen in Figure 3.

PRO Since each portlet have its unique position parameters, this solution removes the need of a separate layouting tool.

PRO The interface and how portlets are moved will still have the same level of intuition as the Absolute position, since it will be based on the same principle.

PRO Works with dynamically sized portlets, without the risk of portlets covering each other as long as the reference points are selected correctly. An example of good behaviour can be seen in Figure 4.

CON Since a relative pointer is to be chosen for every portlet, more user actions are required compared to any of the other solutions.

CON Some users will have problem understanding the concept of relative position, because it is a more complex concept and not as intuitive as Absolute position or Relative position.

CON If the user selects bad reference points, the dynamical behavior of portlets will be bad as well, with a risk of portlets expanding into each other.

CON Editing an existing layout can be quite tedious, because portlets needs not only to be moved, but relative points needs to be reset as well.

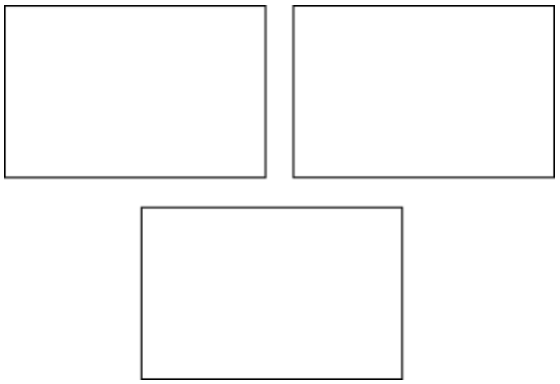

Figure 5: Relative position (auto): Portlets as placed by user.

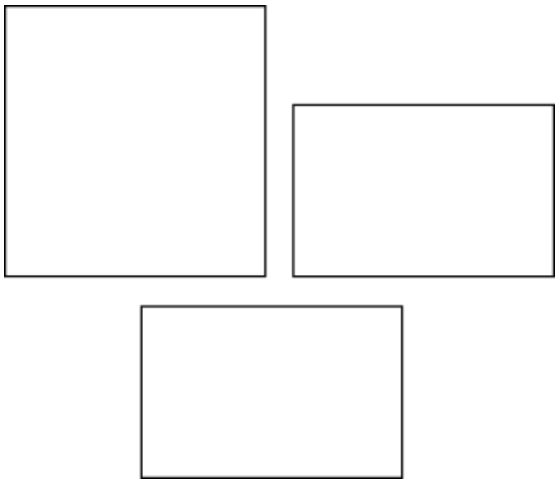

Figure 6: Relative position (auto): One result of automatic declarations with dynamic content.

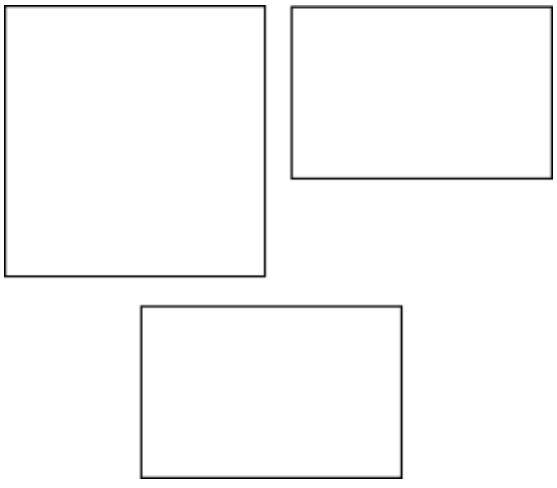

Figure 7: Relative position (auto): Another result of automatic declarations with dynamic content.

4.2.3 Relative position to other content (automatic declaration)

A third solution, based on the previous alternative of relative positioning, but where that the relation declaration is performed automatically. The user specifies the exact position for all portlets, example see Figure 5. Relations are automatically assigned based on the positions. A portlet will be assigned a relation to the closest portlet/portlets in the same horizontal span. Since the portlets may increase their height depending on the content, it is necessary to adjust the position of any portlets that are positioned underneath.

PRO The interface and how portlets are moved, will still have the same level of intuition as the Absolute position, because it is based on the same principle.

PRO Is working with dynamically sized portlets, without the risk of portlets covering each other.

PRO Since each portlet have its unique fix parameters for each page, this solution removes the need of a layouting tool.

CON The behavior of this solution may appear unpredictable, due to what is expected to happen and how the system behaves will vary between users. For example some user will predict Figure 5 to result in Figure 6, if the top left cell would increase in size, while other users will think that the result will be like Figure 7. Since there are several ways the automatic adjustments could be performed, the behaviour of the automatic adjustment could be selected by the user, but then the solution would practically be back to the original Relative position suggestion.

4.2.4 Relative position to layout

Another alternative is to let the portlet placement be defined by a layout. Portlets are placed in areas, which are declared as targetable. The areas that can be targeted and other structural behavior is depending on the defined layout. An example of a possible layout is column structured layout, where each portlet is assigned a column and a position in the column among the other portlets, for example like in Figure 8.

PRO The interface and how portlets are moved have a high level of intuition.

PRO With a layout solution there are no problems with dynamic sizes of the portlets, because the layout arranges the portlets in relation to each other. Example of how the portlets would react in Figure 8 in case the top left portlet is increased in size, can be seen in Figure 9.

CON This solution does not provide the precision that is possible in Absolute or Relative position, since they let the layout define the exact position. However, it is a question wheter this high level of detail control is ever desired.

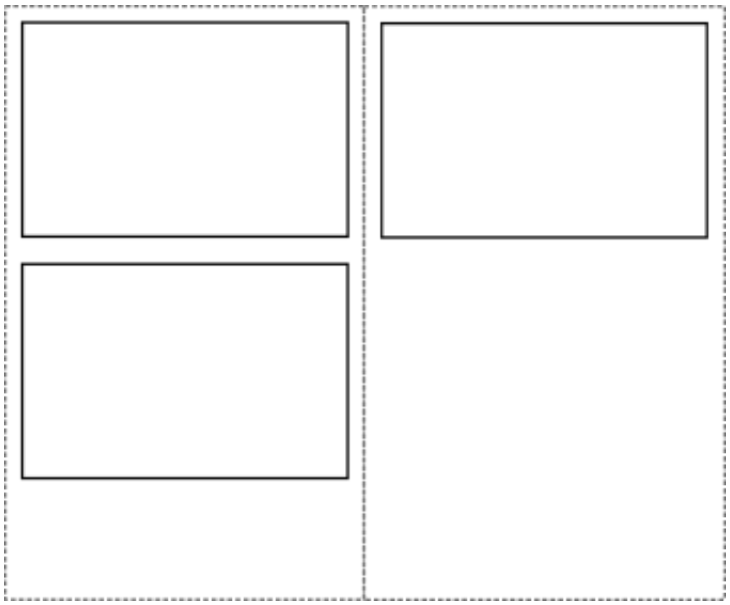

Figure 8: Relative position to layout: Portlets as placed by user.

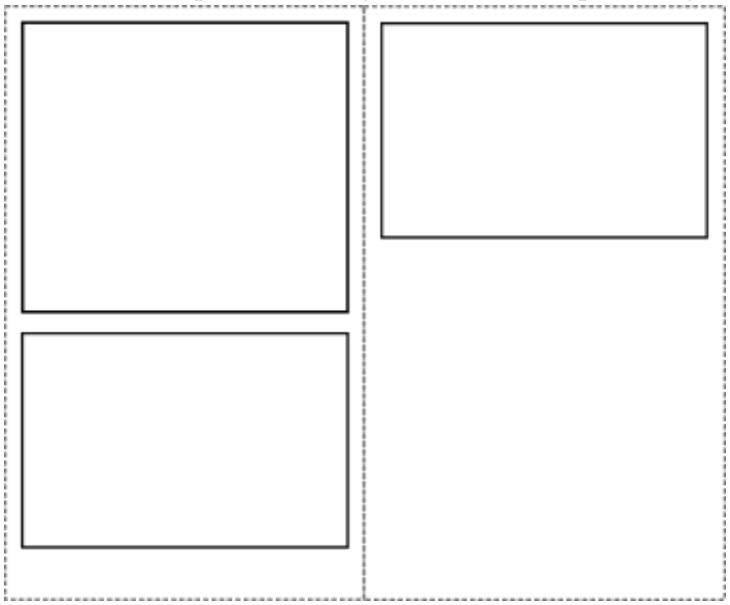

Figure 9: Relative position to layout: Portlets works well with dynamic content.

4.2.5 Summary

Since dynamic portlet support is required, Absolute position lacks of dynamic content support makes it an useless candidate for any further research or implementation of the AeTM. Relative position to content is too complex for some users, if declaration is manual. Relative position with automatic declaration is not intuitive, due to its unpredictable behaviour. All in all, both manual and automatic declaration lacks the require-

ments of being intutive solutions. Relative position to layout is an intuitive, easily used solution, without any considerable risks of misunderstanding. The solution is working well with both static and dynamic portlet behavior. The relative position to layout is choosen as the only one to be considered for any further development.

4.2.6 Placement action

When an object is to be moved, the following actions need to be performed. User need to indicate what object to be moved and where it is to be placed. Since the interface is working in two dimensions, all objects can have their parameters determined in x and y coordinates. Since objects never are to cover each other, a depth relation among the objects are of no interest. Without question, the mouse is the more suitable tool for these kind of actions compared to a keyboard solution.

Since the portlets have a quite big graphical resemblance to a normal window in an operating system, a big set of the users would spontaneously ”grab” the window by holding the mouse pointer over the object and pressing the first mouse button. Holding down the mouse button while moving the mouse, drags the portlet to the desired position. The move action is stopped and the portlet is placed in the last valid position when the mouse button is released.

4.2.7 Inclusion and removal

When used in the application the portlets, that can be arranged, are either placed in the layout or displayed where they are normally. Alternatively, they are not made available in the layout and are then stored as markers in a menu. Markers are text strings or icons that represent the portlets, which may be placed in the layout. When a marker enters the layout, it is replaced with the corresponding portlet. If a portlet is dragged into the marker area in the menu, it is to be replaced by a marker. Portlets vital for the functionality of the application must be available in the layout, before the configuration may be stored. Not desired non-mandatory portlets are left as markers in the menu.

4.3 Layout customization

The user is going to be able to change the layout. The problematics concerning browser behaviour and HTML interpretation are not considered here, but are underlying factors of the described behaviour of the various techniques.

4.3.1 Library

One alternative to provide a layout creation tool, is to provide the user with a large set of pre-made layouts to choose from. By looking at other portlet based applications available on the web, this is the most common solution. In most cases these layouts are a number of columns next to each other with different widths.

PRO Easy for the user to use and understand.

CON Limited to the set of layouts in the library.

4.3.2 Editor (Cell splitting)

The splitter layouting tool is based on diving cells[10] and distributing the sizes. The user selects a cell and have the option to split it, vertically or horizontally. By marking two or more cells next to each other, it is possible to for the user to merge cells back into one cell. Cells that are to be merged, must be placed next to each other. Depending on if they are having a horizontal or vertical relation, they must have the same height or width.

All cells will be potential containers of portlets. The portlets will determine the height of the cell. It is possible to adjust the width of cells by redistributing the width among neighbour cells. The cells have to be next to each other and have the same height, in order to redistribute the width.

PRO Concept is easily grasped by the user.

PRO Any layout could be created with this tool.

CON The user must have a good idea of how the layout should be, since it requires planning to determine in what order the splits are to be performed.

CON Later re-editing of a layout could be quite tedious, because the concept is based on splitting. In order to reach the desired result, this might lead to the user having to merge some cells in order to start a new series of splitts.

4.3.3 Editor (Row and column structure)

Another alternative for layout editing is based on a structure of row and column cells. The cells either hold other cells to define the structure of the layout or the cells are empty and thus targetable for eventual portlets. Row cells are placed horizontally and can distribute their size between them self and their neighbour row cells. Cells can be removed. In case a row cell is removed, the width is redistributed to any eventual neighbour. Column cells are placed vertically and have the width of the parent cell holding it. When a column cell is removed, no redistribution is to be made.

PRO Concept is easily grasped by user.

PRO Any layout could be created with this tool.

PRO Later re-editing of the layout structure is done with ease.

[10] In practice cells are not split, but get half their original size and a neighbour cell of same size is created.

4.3.4 Summary

The library concept is already well known and is used by today's most famous portlet based web services, such as iGoogle[6] and netVibes[8]. The concept is working, but is limited. Since this area already have existing solutions and don't present any subjects that would need any research, no further effort will be made in this area. Since operations will have to be done in a specific order, to gain a curtain result, splitter editor have a concept that is easy to understand, but hard to use. In most cases there will only be "one way" to get a specific result. The Row&Col editor is based on an easily grasped concept. Since new cells with ease can be inserted anywhere later re-editing is easy and a superior alternative compared with the splitter method. The Row&Col solution is therefore the only one to be considered for any further development. A combination of the Row&Col and a library could be a good solution as well, where users can store the customized layouts into a library. But again, this would simply be an library extension to the editor and could easily be implemented without too much effort put into it.

4.4 Look-and-feel customization

The goal of the subject is the research of the possibility of redefining the structure and design of some identified parts of the AeTM-application and to prototype the solution. The parts to be editable are:

- Header
- Footer
- Navigation bar
- Portlet

A description of the different areas is available in 2.2 The AeTM user interface.

Configurable parameters The structure and design of the page is to be configured by changing the HTML-code and CSS-code. This is done by presenting the user with the existing structure in an editor, which can redefine it. The HTML and CSS is only affecting the area that is currently edited, for example may no general CSS for the whole site be changed.

Non-configurable parameters All the identified areas hold content that is vital for the functionality of the AeTM-application. For example all portlets have a title, a minimize button and in some cases a help button. This content is not only vital for functionality of the application, but also have a complex structure and is required to be server side generated. Since these objects must not be removed or altered, but are still to be moved within the structure or design, raise a problem when it comes to user customization. :

The problem is solved by introducing special tags for all mandatory content. The tags are based on the normal syntax of HTML but are no normal HTML elements, that can be interpreted by the browser. Portlet tags could be:

- <portlet_name />
- <portlet_help_icon />
- <portlet_minimize_icon />

For all mandatory content, there will be a corresponding special tag.

Limitations To create a competent graphical HTML and CSS editor with high usability, where there is very low requirements on the user knowledge, as e.g. Macromedia Dreamweaver, would require a tremendeous amount of work. This solution is to be limited to the users who knows HTML and CSS or has to use a separate application, that creates HTML and CSS.

4.4.1 Editor components:

Input areas Two textareas are available for editing the HTML and CSS. When the user starts the editor and selects what areas that are to be edited, the currently used code is loaded into the textareas. The code is edited through the textareas as a basic text editor. There are buttons next to the textareas for creating a preview, submitting the code, exiting the editor, or changing the area to be edited.

Response information There is a separate area, that displays information related to preview or submit requests. Warning if unpermitted parameters occur, mandatory tags are missing, or any other related information. Unpermitted parameters could be Java Script parameters other forms of injections.

Preview design display If the user requests a preview, there is a dedicated area for displaying the resulting graphical representation.

4.4.2 Editor operations:

Preview There is a possibility to request a preview of the inserted code. The code is scanned for any prohibited content and checked, that there are no missing mandatory tags. Warnings or other information is presented in the Response information window and a complete HTML document is composed and displayed in the Design display area.

Submit The user may submit the HTML and CSS for storage. The same checks as for the preview are performed for validity. If the code is accepted, it is stored by the server and replaces the previous structure. In case the code is not valid, it will not be possible to store it on the server, until the malicious code is removed or all mandatory tags are included.

4.4.3 Summary

This solution brings the possibility to define an unique structure and design of the AeTM, where the tradeoff is specific user requirements. The editor meets the requirements, but could be subject of further studies, in order to look at the alternatives of customizing the editor to reduce the user requirements.

5 Results

5.1 General

The research in Portlet positioning and Layout customization covered studies of already existing solutions in the area. A lot of effort was also put into working with new, more elaborate solutions of the subjects, independently of previous existing solutions. The existing available solutions and the new ones were compared, with respect to functionality, usability and possibility to integrate into the existing product.

During the project, three applications which introduced web 2.0 functionality to the AeTM-web application were developed. The applications were based on the most interesting solutions from the Solution suggestion 4 integrated on an instance of the AeTM product. For further information see appendix A.

5.2 Issues encountered during the project

Browser interpretation A lot of problems experienced from testing and development were browser related. None of the major browsers today follow the same standard of how HTML documents are to be interpreted, instead all have their more or less own way to display the result. The differences are especially obvious in how elements are arranged in relation to each other graphically, which is the core of positioning and layouting. Since a goal in the project was to have all applications with A-grade browser support for the major web browsers, more effort than expected was put into studying and working with the different ways, to get the same behaviour out of different browsers.

Row&Column representation problem Since the rows and columns solution was based on the concept of cells holding each other, there was a problem of how to represent this graphically in a good way in the editor. One problem was that finished layout was not going to show the underlying structure of rows and collumns, but at the same time this was necessary to see the structure in order to manipulate it. The final solution was to have two separate modes, where the first one is editing mode and the second mode show the actual resulting structure, see Figure 10 and 11. While in editing mode, all structure cells hold other cells inside their own cell area. In this way, the user gets an idea of how the structure is defined, because the depth becomes graphically represented. Since the top cells of the structure will appear smaller than they will be in the resulting layout, the depth representation is unfortunately a bit misleading. This makes it necessary to have non-editorial representation.

Figure 10: Graphical representation while in editing mode (Gray indicate a row element, black indicates a handle element and white indicate a column element)

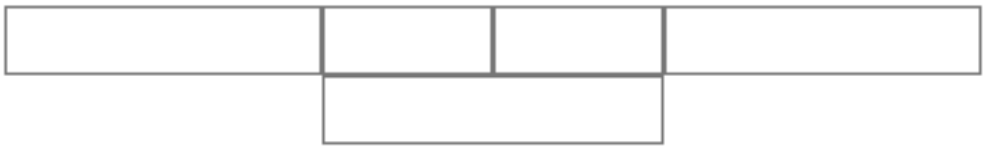

Figure 11: Resulting cell structure (Structure and depth not visible)

6 Discussion

6.1 Evaluate solution

Layouting and positioning The Portlet positioning application combined with the layout customization application introduces new ways of user defined web environments. All big widget based applications, available at the time of the studies are limited to a positioning tool, similar to the one developed during this project, but with very poor layout alternatives. The most common solution is to provide the user with the number of columns to choose from, usually in the interval of one to four columns. How the columns shares the width of the page, is usually fix and can not be adjusted, with the exception for the Netvibes[8] service, which allows the user to distribute the space among the columns.

The applications of positioning and layouting is extending the possibility to let the user define unique layouts and positions of the portlets. The layout tool introduces a new concept, to create a structure with support for dynamically sized content. The position tool is not introducing anything new. The drag and drop of HTML elements was introduced a couple of years ago, but never the less it is vital for the functionality and usability of the application.

Testing and reviewing by the AeTM User Experience team determins, that there are areas where the applications can use improvements, but that the over all concept is interesting. Main areas of improvements concerns introducing corresponding animations for all actions, making the reaction of an action clearer. Other areas of improvement are

of a pure aesthetic nature.

Look-and-feel customization The look-and-feel prototype allows the user to redefine the structure and design of some identified parts by editing the HTML and CSS. The solution is a successful compromise between user configuration possibilities and application reliability. Since all parameters that held any functionality were contained in mandatory tags, the user have had a lot of freedom while using the editor, yet without the risk of compromising functionality.

Testing and reviewing with the AeTM User Experience team determins that the application is an easily used editor, for a user who knows how to write HTML and CSS and that it holds the desired functionality.

6.2 Limitations of solution

Adding new functionality to AeTM The AeTM is of a static nature, where all customers have more or less identical instances of the product, with the exception for community logotypes. In this state, adding or removing portlets and links or making other changes affects all communities in the same way. The development team can thus easily make changes to the application.

A question that arise - since the possibility to have uniquely defined communities - is how new content is added and inserted in all communities. Since the communities- after implementing the suggested changes from the project- all can be unique and thus adding new content will practically be impossible. A possible solution can be that the administrator defining the layout and customizing the structure, declares suitable areas where new content can be added. Preferably all layouts will have a mandatory portlet, that are required to be placed in a cell for incoming portlets. All identified areas will also require mandatory tags to indicate incoming new content. Not enough effort is put into resolving this issue, thus there might be better solutions then the one suggested.

User requirements Since the user is required to know how to write HTML and CSS, the look-and-feel application have a limited user base. In order to broaden the user base, there is room for a lot of enhancements in this area. Though in the end it will practically be a question of implementing the features of today's normal HTML and CSS editors into the web editor.

7 Conclusions

7.1 Most important results

A new way of defining layout The most interesting result was the final solution of the layout application. Since the application manages to bring the user a new and intuitive way of layout construction. During the pre-studies, no other solutions similar to the kind created in the project were encountered. The widget based services found, which allowed the user to alter the content, were all limited to either a static layout or a layout where the user could choose from the numbers of columns the layout should have. Thus this solution actually introduce a way to allow more complex user defined layouts with dynamic content.

7.2 Recommendations of future development

HTML & CSS editor The editor for changing the HTML and CSS, for some identified areas, have much potential for further development. The work would involve introducing an interface that removes the user requirement to know HTML and CSS. For future development it would also be possible to consider to also allow the user to provide own content, not limited to HTML and CSS.

Layouting and positioning For the layout and positioning applications there is some smaller areas, where there is possibility for future development. Mainly cosmetic related questions, but also to work with the applications displayed confirmation on user input actions. The user must be given a response for all input actions, so that there is no hesitation whether the user input was interpreted by the application or not.

7.3 Summary discussion

This project was a great experience and I have learnt a lot during the short period of time I have been working on it. Of course I have been learning a lot about practical software development, for instance how bad design choices tend to haunt you later. Bad design choices usually took longer time to go back and change, rather than it would have to take the time and think through the design properly.

Estimating the time that would be required was also not always that easy. A lot of time I discovered that the estimated time was not enough, especially later in the project when it came to writing the code. Usually if the estimated time was not sufficient it would be due to a problem that occurred that was unpredicted, and not due to the work pace. By later adding some buffer time, the time taken would correspond better with the estimated time.

8 Terms

Global Distribution System, GDS
e-Travel Management web application, AeTM
HyperText Markup Language, HTML
Cascading Style Sheets, CSS
JavaServer Pages, JSP
Java Scripts, JS
Asynchronous JavaScript and XML, AJAX
Portal, major web application framework
Portlet, smaller web application similar to widgets and gadgets
User interface, UI

References

[1] Yahoo! developer network , *The Yahoo! User Interface Library (YUI)*, 2008
`http://developer.yahoo.com/yui/`

[2] Yahoo! developer network , *A-Grade Browser Support*, 2008
`http://developer.yahoo.com/yui/articles/gbs/index.html#gbschart`

[3] Craig Larman, *Agile and Iterative Development: A Manager's Guide*, August 11, 2003
ISBN-10: 0-13-111155-8

[4] Albert N. Badre, *Shaping Web Usability: Interaction Design in Context*, January 23, 2002
ISBN-10: 0-201-72993-8

[5] Jakob Nielsen, *Usability Testing With 5 Users*, March 19, 2000
`http://www.useit.com/alertbox/20000319.html`

[6] Google, *iGoogle*, 2008
`http://www.igoogle.com`

[7] Google, *Google docs*, 2008
`http://docs.google.com/`

[8] Netvibes, *Netvibes*, 2008
`http://www.netvibes.com/`

[9] Snap Group, *SUMO Paint*, 2008
`http://www.sumopaint.com/`

9 Appendix

A Design document

B Investigation and development of new Web features

C Training objectives

Design document

LAYOUTING, CONTENT POSITIONING & CUSTOMIZATION

Samuel Svensson

Contents

1 Introduction

This document will go through the design and structure of a set of applications for extending the customization possibilities for a web application. The goal of the interfaces are to make intuitive and easily used tools without any risk of compromising the high detail configurability. These applications shall be available through any a-grade browser[1]. Without the need of any external and, separately installed software, such as for instance the Flash player. The applications will be intregratable with the Amadeus e-Travel Management web application, after this referred to as AeTM-web, with non or limited change to the existing product.

2 About the interfaces

There will be several interfaces. One interface is creating and modifying the layout, how the area should be split up and divided. Another interface is for insertion and positioning of the content of the site. Other interfaces are configuration of a more detailed graphical nature and also solutions where the user can bring desired content to the application.

3 About the AeTM-web

The current AeTM-web product is today based on the BEA WebLogic Portal. This application uses some concepts which these web interfaces are based on. The main subjects which touch this assignment are portals and portlets. A portal is a major framework which in its turn handles various portlets. The portlets are more or less complex programs similar to the today more well known web-widgets or gadgets, but the portlets can also have a way more complex functionality compared to the normal widget.

[1] To have A-grade browser support means to provide identical experiences for the user, independently of the chosen browser software.

4 Design of user interface

4.1 Layouting

Layouting is a tool for creating and modifying the structure of a page. The structure is presented to the user, who can select any part of the structure and choose a modification action from a menu. Selection and menu choices are all done through mouse actions.

4.2 Portlet positioning

Portlets which may be added to a page will be available through a menu. A portlet is added by being dragged into the layout, from the menu and dropped in a targetable area. A portlet placed in the layout may be moved around by dragging and dropping between different target areas. Portlets may also be removed from the layout by being dragged back onto the menu.

Portlets made available in the layout will be rendered as they normally appear in the AeTM-web application in order to give the user a fairly good idea of what the result will look like.

4.3 Look-and-feel customization

A look-and-feel tool shall be implemented to make the user able to do some customization of identified parts of the portal. The identified parts of the application are:

- The header
- The footer
- The navigation bar
- The portlet frame

The tool will bring the possibility to configure style sheet parameters as well as the actual Hypertext Markup Language structure, after this referred to as HTML structure.

5 Preliminary Design

With respect to given requirements previously mentioned in the introduction, the interface itself will be based on HTML. In order to be able to have an interactive web interface, without necessarily reloading the page for each action and having very limited functionality, it will be necessary to extend the HTML with JavaScripts.

5.1 Layouting

The layouting tool will be an application with which the user can create a dynamic and scalable grid of cells, where portlets may be inserted. The layout will be based on cells of either row or column type. A cell may hold two or more cells, of the opposite type of its own, or it may hold one or more portlets. There can for instance never be a cell of row type containing other cells of row type, it can only hold either portlets or column cells, unless it is empty. Thus cells can be considered to be either structure cells when holding other cells, or containment cells for holding actual content. By selecting a cell with a mouse click through the interface, the user will be presented with a couple of alternative actions for adding new cells or removing cells. Actions possible to a cell will be:

- Add children cells of the opposite type to the selected cell
- Add new sibling cell of the same type before selected cell
- Add new sibling cell of the same type after selected cell
- Remove selected cell

Some actions performed to the layout might break the rules of the structure, such as a cell should never hold another single cell, which occurs if one out of two cells are removed. Therefor there is a need for a simplification function that adjusts the structure and removes the other cell as well and then moved any eventual content. If a cell is of the row type, it will be possible to modify the width. This will be done by all row neighbour pairs will have a common handle, which by holding mouse button down and dragging will redistribute the size between the two neighbour cells. When the whole layout and all positioning steps are done, all cells will automatically adjust their height after the content. A cell holding one portlet will have the height of the portlet and an empty cell will have a height of zero units or a minimum set height. This is the reason for only having handles for row elements because when it comes to cell width it is the other way around, since the width of the cell is determining the width of the portlet. Each individual cell in the layout structure will need to hold the following data:

- If the cell is a row or a column.
- An array of all children cells inside the current cell.

- Two pointers to the next and previous sibling cell, if they exist.

If the cell is of the row type, some additional information will be necessary:

- Two pointers to the left and right handles of the cell, if they exist.
- The area the cell should take up in within the cell in percentage.

Figure 1: Illustration of cell relations where gray indicate a row element, black indicates a handle element and white indicate a column element. The left and right row cells are empty, the middle row cell holds two column cells where the top column cell holds two empty row cells and the lower column cell is empty.

In order to be able to tell which cells are holding each other there will be some empty space between the child cell and the parent cell. This is only going to be the case when creating the layout, the resulting structure will fill this space by increasing the child cell area and remove all handles.

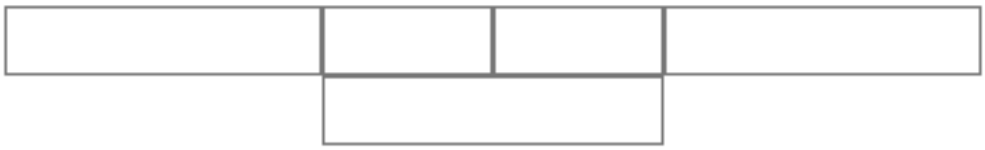

Figure 2: Resulting cell structure from example in figure 1. The structure itself will in almost all cases be invisible in the resulting page, but will set the rules for the content.

5.2 Portlet positioning

The positioning tool will be used after, or simultaneously with, the layouting tool. Available portlets will be presented in a list represented by their name in text, or if they are positioned in a cell in the layout they will be shown graphically. Portlets can be positioned by selecting the object with the mouse and dragging it to the desired cell, or back

to the portlet list. For positioning portlets, all cells that are to be considered targets for portlets shall be empty and may not hold any children cells. The list where available portlets are stored in text form should be considered a target for portlets that are not wished to be available at the site, all portlets placed in the layout should be able to be returned to the list. If a portlet by dragging enters an empty cell, the portlet will be placed there as a preview. On release of the mouse button the drag-action will stop and the portlet will be placed in the most recently legit target visited. Each individual cell that holds no children cells will only need to hold the following data:

- If the cell should be considered a empty target or if it already has been targeted by a portlet.

There may be more than one portlet per cell, but after the first portlet is placed in a specific cell the following portlet will be placed in relation to first portlet which ignores the cell thus actually behaving independently of the cell itself. In case a portlet is to leave a cell empty, the cell will be declared as a target.

5.3 Look-and-feel customization

The look and feel customization tool will bring a high level of configurability to the user. By giving the user the possibility to edit the HTML and CSS of all the identified components. Both the HTML and CSS are accessed in a respective text area, where text can be added or removed like a basic text editor. A preview of the result is presented in a separate frame, the preview is updated on request from the user. In editing mode the user will have the following actions:

- Manual refresh of the preview.
- Store the current layout.
- Exit the editor.

All the identified parts that can be customized have elements that are necessary for the functionality of the application. The user is going to be allowed to decide where the mandatory elements are to be placed in the HTML, but will not be privileged to modify its content beyond graphical parameters in the style sheets. Mandatory content will be included in special tags and on a requested preview the tags will be replaced with the corresponding content. There will also be checks performed to make sure that the user have not forgotten to include the mandatory tags. I the case a tag is missing, the user is informed by a message and the code may not be stored.

The user is informed of which the mandatory tags are at the same page as the editor.

6 Detailed Design

Since the application is browser based, all elements visible to the user are Document Object Model-elements. The DOM is a standard object model for representing HTML or XML, it is platform and language independent. All cells, handles, portlets and markers are generic block level containers of DIV type.

A DIV may be given a large set of graphical parameters to describe how it should be shown and behave in the browser. These parameters may either be set directly in the HTML for every element or be set as rules for a larger set of elements. The styles are either written directly in the HTML or in a separate style sheet. The DIV element is a container and may hold any element itself.

6.1 Layouting module

6.1.1 Cell object

The cells of the layouting tool have a need for the following set of variables to be able to coop with the demands of functionality in the preliminary design. Each cell should hold the following variables:

isCell To declare the element as a cell.

isRow / isColumn All cells need to hold information whether the cell is of row or column type.

isTarget If a cell is empty it should be considered a target for portlets, if it holds children cells or portlets it should not be targetable.

isSelected To be able tell if the current cell have been selected or not.

realChildren A set of all children cells.

parentNode The parent cell.

nextCell The following sibling cell in layout.

previousCell The previous sibling cell in layout.

If the cell is of row type the following variables should be held as well:

leftHandle The handle on the left side.

rightHandle The handle on the right side.

relativeSize The relative size. Float number to represent the width the cell should take up.

6.1.2 Handle object

The handle that redistributes the width between the cells hold the following variables:

isHandle A variable declaring the element as handle.

leftCell The cell on the left side of the handle.

rightCell The cell on the right side of the handle.

6.2 Layouting functions

6.2.1 Cell

select When clicking on an unselected cell it becomes selected. One cell can only be selected at the same time, so on selection of a cell any other selection is removed.

deselect When clicking on a selected cell it becomes unselected. Also called by select when more than one cell is selected.

addCell Insert sibling cell of the same kind before or after selected cell.

addChild Insert child of opposite type in the selected cell. If the cell is empty, two cells are added. If the selected cell is holding a portlet, the content is to be inserted into the first of the two new children.

removeCell Selected cell is removed, any cells held by this cell are removed. Any portlets are moved to the portlet list. When cells are removed and rules of the structure have been broken the simplification function is called. If all cells are removed from the layout the user will be asked to insert two row or column elements.

simplification Corrects any structure errors. For instance a cell without siblings is removed and all its content is moved up one level in the tree structure.

6.3 Layouting events

6.3.1 Handle

When the user interacts with the draggable elements a lot of events are fired. During one drag session, the following actions are being performed for the different events, for the handle movement.

onDrag Depending on the mouse movement in X-axis, the width is redistributed between the two neighbour cells. If the handles dragged to the left the size of the left cell is decreased and the right cell increased and vice versa. A minimum size of both cells are taken into account, so the handle stops the distribution as one of the two cells is to be smaller than the minimum width.

onMouseUp When the mouse button is released and the drag action is finished, the relSize variables for the two cells are calculated from their new widths.

6.4 Portlet positioning modules

6.4.1 Portlet object

The portlet will need to be represented graphically and in text form. For each portlet there will be a marker, which represents the portlet in a more convenient way as a text string. Variables for the portlets:

marker The marker that represents the portlet in text form.

target The current target the portlet is related to.

6.4.2 Marker object

Variables for the markers:

portlet The portlet that the marker represents.

target The current target the marker is related to.

6.5 Portlet positioning functions

6.5.1 Portlet & Marker

switch Since the marker is a different graphical representation of the portlet depending on the context. There will be a switch function that converts the portlet to a marker

and vice versa. This will be used when a portlet is moved from the layout to the portlet menu or a marker is placed in the layout from the portlet menu.

6.6 Portlet positioning events

When the user interacts with the draggable elements a lot of events are fired. During one drag session, the following actions are being performed for the different events.

startDrag When the marker or portlet is starting to get dragged a proxy element is created to represent the dragged object, the proxy will follow the mouse movements. The proxy has the same size as the element with a border but is otherwise transparent. The element becomes a little bit transparent to indicate that it is to be moved.

onDrag When a drag action is performed, the direction of the drag in Y-axis is being determined for every movement and is later used by the onDragOver.

onDragOver When the pointer enters a targetable cell during the drag action, the element is moved to that position. If the moved object leaves the previous cell empty as it leaves, that cell becomes targetable.

When the pointer enters another draggable object during the drag action, the element is either positioned over or under that object, depending on the direction in Y of the drag. If the mouse is going up, the inserted element will be on top, otherwise it will be inserted below.

The menu holding unplaced markers is to be treated equally to a cell object when it comes to targetability for dragged objects.

endDrag When the dragging action is complete the proxy element is removed and the moving element get its normal transparency back.

6.7 Look-and-feel customization events

6.7.1 Preview

When the user requests a preview, the HTML and CSS in the text areas are composed as one HTML document, where the CSS is included as a style parameter in the beginning of the document. The document is loaded in a separate frame and overwrites any previous content. Any mandatory tags are replaced with the corresponding content.

When the content is being read from the text areas it is checked for prohibited content. If illegal content is encountered, the user will be presented with a message that not allowed content have been found and will have to be removed. It will still be possible to preview the content but not to store it.

For a detailed description of prohibited and mandatory content see 6.7.3.

6.7.2 Store

When the user request to store the content, HTML and CSS are sent to the server which checks that no mandatory tags are missing or that there is no prohibited content. If the code breaks any of the rules the user will be informed what is wrong with the submitted code and the code will not be stored. If the code is accepted the mandatory tags are replaced by real content and saved as server side code. The HTML and CSS will be saved as well, but will not be used unless the user chooses to use the look-and-feel tool again.

For a detailed description of prohibited and mandatory content see 6.7.3.

6.7.3 Content control

When a user chooses to preview or store, the HTML and CSS will be validated to not miss any mandatory content or hold any content that could be malicious code. The check will be done with regular expressions, and will check for:

- That there is exactly one encounter of every mandatory tag.
- That there is no occurrence of JSP tags.
- That there is no occurrence of SCRIPT tags.
- That there is no occurrence of DOCTYPE tags.

Investigation and development of new Web features

Description
The goal of the project is to study and propose some alternatives and enhancements to the current design of AeTM Web product. The overall task is to create a solution for easy web designing through the browser itself, where the user should have no required knowledge about web designing or coding at all. With the vast quantity of elements that should be configurable through this interface the challenge, and where a lot of focus will be, is on how to make a clear and easily used interface without compromising the configurability.

The Amadeus e-Travel Management product is a booking solution for companies which have employees that need to travel in their work. The company books the trip for the employees, or letting the employees book themselves.

Where and how
The trainee will be integrated in a team of 10 persons whose responsibility is to develop User Interface of e-Travel Management (AeTM) Web product. The project will be done at Amadeus in Sophia Antipolis in France. Supervisor for the trainee will be Patrick Chalony.
The internship started the first week of February, and will stop in the end of July.

Task description
The goal of the project is:

- Research and evaluation in order to determine suitable tools and techniques.
- Model solutions for the interactive web interface.
- Evaluate and test solutions.
- Implement solution for the AeTM Web product.

The main focus will be researching and modeling a suitable solution, thus the major problem will be the human interaction aspect with respect to configurability and ease of use. Several aspects could be considered:

- Page layout
- Content aggregation
- UI customization

Rough schedule
→ Feb-Mar Analyze and prototype a solution for customizing page layout
→ Apr-May Propose solutions for integrating third content
→ Jun-Jul Study what exists today for UI customization (for example, the Facebook Markup Language FBML, etc.)

amadeus
Your technology partner

Memo

To: Patrick CHALONY

CC: Samuel SVENSSON

From: Hélène PASERO

Telephone:

Department:

Date: 26/02/2008

Subject: Training objectives

The objective of this internship is to study and propose some alternatives and enhancements to the current design of AeTM Web product.

In order to cope with the current industry trends and the soar of Web 2.0, we have to keep improving the UI and developing some enhanced features that will address some usability issues or improve the general look-and-feel of the application.

Here the focus will be more given on customization strategies, and will contain feasibility studies, propositions of ways to improve customization and some development to assess the propositions.

Five types of studies have been identified for the training, ordered here by level of significance.

- **Portlet display and positioning :** AeTM product is based on portal technology, and each logical module of one given page is displayed inside what we call a portlet. The portlet positionning and their display is currently defined during product installation or based on some parameters evaluated in the application. The goal of this study is to enable the user to decide dynamically the positioning and the display of one given portlet (select and move a portlet with the mouse, place the portlet at some authorized locations predefined in the layout). If possible, the study can include the possibility of persistency.

- **Layout customization** : Give the user the possibility to define his/her own layout for one given page. This study will include a list of propositions on what is generally expected for layout customization, and some feasibility studies.

- **Look-and-feel customization :** Give the user the possibility to customize the look-and-feel of some idenfied parts of one page. A page is composed of a header, a footer, a navigation bar and the central part, composed of many portlets. The goal here is to study the possibility to upload some html/css code to customize the header/footer/navigation bar and apply a personal skin on some portlets.

- **Content personalization :** The portlets currently displayed to the user are only the ones defined by the product. The goal here is to be able to create new types of portlets, where the user will be able to insert personnalized content : either portlets containing frames where the user can redirect to other urls, either portlets with RSS content, either portlets containing uploaded html and corresponding CSSs.

- **Skin online creation :** AetM product currently offers the choice to apply different skins : one skin is applied by default, and some others are created by our team on customers demand. As the process is centralized within our team, it lacks of flexibility. The goal here is to think about a new module in the application through which the user would be able to create his/her own skin online and make it persistent.