Where Web Engineering Tool Support Ends: Building Usable Websites[*]

Richard Atterer
Media Informatics Group
Ludwig-Maximilians-University Munich, Germany
richard.atterer@informatik.uni-muenchen.de

ABSTRACT

In this paper, two of the currently available Web Engineering solutions (UWE and OO-H) are analysed with regard to the question whether websites created with them and their tools have a high usability. Additionally, the respective models are examined to see whether usability aspects can be expressed with them. In a small case study, an example website is created by converting a model to an implementation manually. Special attention is paid to usability issues regarding both the generated pages and the development process. Subsequently, the manual conversion is compared to a tool-supported process.

Categories and Subject Descriptors

H.5.4 [**Information Systems**]: Information Interfaces and Presentation—*Hypertext/Hypermedia*; D.2.2 [**Software**]: Software Engineering—*Design Tools and Techniques*; I.6.5 [**Computing Methodologies**]: Simulation and Modeling—*Model Development*

Keywords

Web Engineering, usability, modeling, comparison study

1. INTRODUCTION

Creating an easy-to-use user interface is just one of the many challenges that the developers of a web application face during their work. Due to the many factors which influence the user experience of the application (ranging from differences in the user's input and output devices to latency issues and the request/response paradigm of the HTTP protocol), an application with the same functionality typically has a quite different interface when implemented as a web application instead of a GUI application for a desktop computer.

[*]This work was supported by the BMBF in the context of the intermedia project, `http://www.intermedia.lmu.de`

Current Web Engineering approaches such as UWE (UML-based Web Engineering, [6]) and OO-H (Object-oriented Hypermedia, [3]) try to offer a complete solution to generate web pages (or assist in their generation), typically providing models for the application logic, navigation structure and the final presentation of the web pages.

In this paper, the focus is on the navigation and presentation aspects, in particular the question whether the output of the existent tools results in websites with high usability. For the two Web Engineering solutions mentioned above, the following is examined:

- Does the *method* involve aspects which have the goal of improving usability?
- Do the navigation and presentation *models* allow expressing usability constraints?
- Do the *tools* have support for what the methods/models provide in terms of usability?

This paper is structured as follows: Section 2 introduces the example setting of business processes in a travel agency and describes the experience of creating a website prototype supporting these business processes, in particular the issues relevant to usability. Subsequently, section 3 is concerned with the existing solutions UWE and OO-H and the usability support they provide. Section 4 discusses perceived shortcomings with the current methods and tools, and section 5 presents some conclusions and an overview of possible future work.

2. CASE STUDY

The entire process of modelling and implementing a website was performed with a small example. All steps were deliberately performed manually in order to get a feeling for all the issues related to creating a working, usable website from nothing more than an idea of the business processes it has to support. A rough prototype was implemented in PHP.

In the search for an example setting, the companies that are the focus of the intermedia research project appeared a little too complex, so an example for an information-based business outside the usual industry branch of the project was selected. (The intermedia project is concerned with the analysis of intermediaries in the media industry – companies which deal with or forward information, like book publishers, music labels etc.)

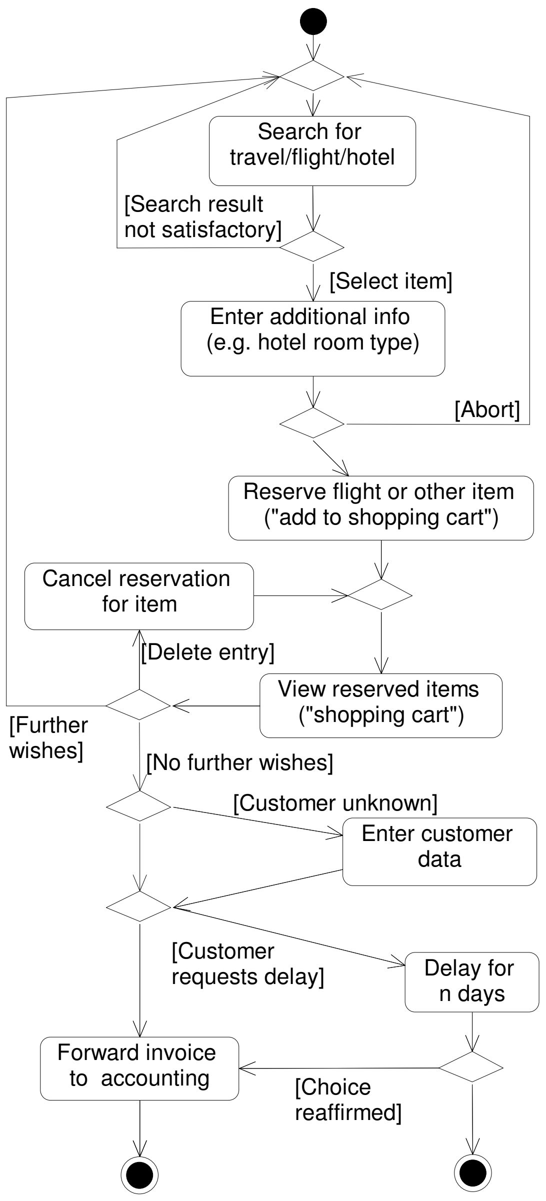

Search for travel/flight/hotel
[Search result not satisfactory]
[Select item]
Enter additional info (e.g. hotel room type)
[Abort]
Reserve flight or other item ("add to shopping cart")
Cancel reservation for item
[Delete entry]
View reserved items ("shopping cart")
[Further wishes]
[No further wishes]
[Customer unknown]
Enter customer data
[Customer requests delay]
Delay for n days
Forward invoice to accounting
[Choice reaffirmed]

Figure 1: Activity diagram for a business process: Searching for and booking travels in a travel agency

2.1 Example Setting: Travel Agency

For the subsequent work, the example that was chosen is a travel agency. This choice is particularly interesting for usability research because of the following properties:

- Several quite *different activities* need to be combined in an intuitive way in the same user interface: Searching for information (on travels, flights, car rental etc.), collecting "interesting" items and booking/reserving/ cancelling items.

- The same interface (with only minor variations) should be usable by *different audiences*, i.e. both by customers booking flights from home and by travel agents serving customers in the agency.

- The setting mostly fits into the standard, widely-used application type of an "online shop", so any conclusions drawn will be applicable to a large number of existing web applications.

2.2 Creating the Website

Going from the mere idea of the business to be modelled to the finished website involved the following steps: Modelling the application, creating a page design and graphical design, and converting the model elements into elements on the web pages.

Modelling [9] describes a systematic approach for modelling business processes as well as analysing and optimising them. This approach was followed to create UML activity diagrams for the example by identifying the business goals and active business partners, determining those business partners' business use cases, describing the different use cases with a few sentences, and finally by modelling the business processes in detail.

Figure 1 shows a simplified version of the UML activity diagram for the main business process, which includes searching for information, selecting items (travels, flights, hotels) for booking, viewing and modifying the list of selected items, and optionally putting the booking process on hold for a few days if the customer wishes it.

At this point, a number of notes regarding the usability of the process can already be made. Ideally, to allow automatic processing by tools, we would like be able to integrate the following points into the diagram in a format other than UML comments:

- Searching, reserving and booking are too intertwined to be separated from each other: During a typical customer visit in a travel agency, the customer will alternately express his wishes, ask for temporary reservations (put a seat on a plane on hold) and ask for more details or alternatives to the reserved items. The web pages should support such quick switches between activities.

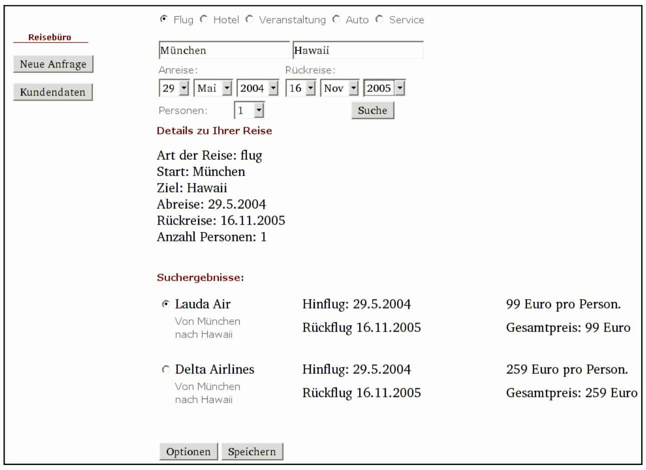
- Searching is the most important activity: It is desirable to have the search facility available on all pages for quick access, and to have a way to remember earlier queries.

- The whole business process may take several days to complete. As a result, the user interface needs to be able to track many concurrent instances of the process, similar to a trouble ticket system.

- Unnecessary manual entry of customer data should be avoided, it should be possible to search for existing customers by customer number or name.

As we will see later, some of these issues can be expressed by Web Engineering solutions, though not necessarily at this level of abstraction.

Page design Once the business process was defined, an overall page design was created for the web application, and the required page elements were fit into it. Whereas the earlier modelling work followed a top-down approach (from



[✓ Flug] Hotel Veranstaltung Auto Service

München | Hawaii

Anreise: 29 Mai 2004 Rückreise: 16 Nov 2005

Personen: 1

Details zu Ihrer Reise

Art der Reise: flug
Start: München
Ziel: Hawaii
Abreise: 29.5.2004
Rückreise: 16.11.2005
Anzahl Personen: 1

Suchergebnisse:

[✓ Lauda Air] Von München nach Hawaii	Hinflug: 29.5.2004 Rückflug 16.11.2005	99 Euro pro Person. Gesamtpreis: 99 Euro
Delta Airlines Von München nach Hawaii	Hinflug: 29.5.2004 Rückflug 16.11.2005	259 Euro pro Person. Gesamtpreis: 259 Euro

Figure 2: Screenshot of the prototype website displaying search results. From the navigation in the top left corner, new business processes can be started. The list of active processes (each holding one or more reserved items) is displayed below it, unless it is empty (as in this case).

an abstract business idea to relatively concrete activity diagrams), this can be viewed as a bottom-up approach.

One reason for this approach is that the low-level page design is actually very standardized for most web applications: Among others, Nielsen [7, Nr 7] [8] has conducted research which shows that a standard page design with navigation at the left, advertisement at the top (if necessary) and other similar items is desirable, because it reduces the time needed by (notoriously impatient) users to learn how the site works.

In the example, a simple navigation area on the left contains links to support the basic functionality, which is to start actions corresponding to the different business processes (customer database, information and booking etc.). As noted in the previous paragraph, the system needs to keep track of a number of separate instances of business processes which run in parallel; they are also listed in the navigation area. See figure 2 for a screenshot of the prototype.

Several typical web application patterns [3, section 3.2] were discovered with the example application: If larger amounts of information, such as customer data, are to be entered by the user, a *guided tour* (pages with "previous"/"next" buttons) is appropriate. A *shopping cart* is also present – the cart is called "reserved items" on the web pages to stress the fact that e.g. a seat on a plane is temporarily "locked" while it is in the set of reserved items. Standard *indexes* are used to create lists of links to search results or active business processes.

In addition to these, a new concept was identified which we will call *inlining*, an allusion to inlined C/C++ functions: Since searching is a central activity for the example, having the user follow links to the search form is not ideal. Instead, a small version of the search form is displayed in all the places where a link to the form would otherwise have been placed. With the prototype website, there is no distinction between the normal and small versions of the search form – the form from figure 2 simply appears at the top of all pages where the search facility should be available.

Graphical design The graphical design was not a primary concern when implementing the prototype for the example website – the resulting pages are simple and only use a minimum amount of graphic elements.

However, in general the graphical design is an important aspect which can significantly influence the overall usability of the final website. The web application developer should take care to add graphics which are not only visually pleasing, but which also do not surprise or mislead the user (e.g. graphics which are not recognisable as links). Instead, the graphical design should stress the intended use of the different elements on the page.

Conversion of the model The model in figure 1 is slightly too abstract to be converted into web pages. For Web Engineering solutions with tool support, the information for automatically generating the link structure and page content is available in the navigation and presentation models, but no such models were created for the prototype. Instead, the activity model served as a basis for the ad-hoc manual creation of "appropriate" links and page content.

In practice, this worked very well – in particular, since only one model is present, you do not need to switch between the "application logic", "navigation" and "presentation" views of the web application, and the different diagrams for these views do not need to be kept in sync. Also, it is usually easy for a human to come up with appropriate page content for a given model; as long as no automatic creation of pages takes place, additional models may just increase the amount of modelling (i.e. the amount of work) without significant benefits.

Consequently, the use of fewer different types of diagrams can be interpreted as improving the usability *of the development process*. It may be advisable to restrict the use of additional diagrams to particularly challenging areas of the application. This would be analogous to current practice in classical Software Engineering, where class diagrams are the predominant form of diagram, and e.g. object diagrams are used to model a small number of selected details about the system.

Further work will be necessary to determine whether the positive aspects of advanced tool support (with automatic page generation) and of having fewer types of diagrams can be combined.

2.3 Lessons Learned

Usability is multi-facetted, it is influenced by aspects at every abstraction level, from the business process model to graphics design. In order to be suitable for creating usable websites, a Web Engineering solution needs to take all these aspects into account, which is far from trivial.

Usability is difficult to get right and easy to get wrong. Since most people designing web applications are not experts in this field, it would be nice to have more than just usability guidelines like [2] and [7] – such guidelines need to be integrated into the design process, and ideally even the Web Engineering tools.

3. A LOOK AT EXISTING SOLUTIONS

In this section, we will have a look at the navigation and presentation aspects of UWE (`http://www.pst.ifi.lmu.de/projekte/uwe/`) and OO-H (`http://gplsi.dlsi.ua.es/iwad/ooh_project/`), and the respective Web Engineering tools, ArgoUWE and VisualWADE, to check how well the creation of usable websites is supported. As mentioned in the introduction, the method, the models and the tools will be examined.

3.1 OO-H

The Object-oriented Hypermedia Method is described in [3], more details on presentational aspects are available in [1]. Creating a web application using OO-H involves creating standard class diagrams and, based on these, *navigational access diagrams* for each type of user. Using these navigational diagrams, a default web interface, intended for quick prototyping, can be generated.

For more sophisticated web pages, a default *abstract presentation diagram* is derived from the navigational diagram, and subsequently refined by the web application developer. Essentially, the presentation diagram represents a template mechanism.

The VisualWADE tool offers very good support for all steps of the development process, including automatic generation of prototypes.

This approach is very general and leaves the developer free to add steps like usability testing with prototypes.

One item that is directly relevant to usability is OO-H's decision to promote the generation of different navigational diagrams for different types of users. It is not clear whether this is supposed to include not only descriptions of the same process from different perspectives ("customer books seat on plane" vs. "airline executive accepts booking"), but also by different audiences ("customer books" vs. "travel agent books"). The latter adaptation of content must be performed with great care – in particular, websites which present different content after asking the visitor what audience group he belongs to (e.g. for B2B, "home office" and "small company") can be frustrating to use.

The method's use of certain patterns is laudable. The available patterns deal with some problems related to usability (such as the user's tendency to "get lost" on sites with a large number of pages) and try to provide standardised ways to deal with the problems (e.g. the *Location Pattern*, which adds headers and footers with navigation information to pages).

The use of patterns is a step in the right direction, but in the author's opinion, the available patterns are very concrete, and the whole presentational diagram could easily be replaced with a more traditional template solution such as SSI (server-side includes). More abstract usability guidelines, e.g. "searching is a central activity and needs to be available everywhere", cannot be expressed.

Because automatic creation of prototypes is possible, the models given as input to the tools must describe the web application in appropriate detail. This means that a significant amount of modelling is necessary to get relatively simple results – this makes the development process somewhat tedious and time-consuming.

3.2 UWE

The approach of UML-based Web Engineering, described in [6], is comparable to that of OO-H. After creating the conceptual model, which describes the application logic in the same way as in classical Software Engineering, *navigation* and *presentation models* are constructed. These models are subsequently converted into an XML format using the UWEXML preprocessor. Next, the UWEXML generator semi-automatically produces XML templates, presentation stylesheets etc. The developer can manually refine the output to suit his needs. In the presentation model, patterns like a *guided tour* (pages with "previous" and "next" buttons) or indexes (lists of links) can be utilised – similar to OO-H, the presentation diagram is essentially a template mechanism with support for patterns.

In [4], the method is described with a focus on the user interface and extended with the aim of improving usability. Storyboarding and the creation of pure HTML prototypes are introduced to help the developer with the design of the presentation model.

The tool support for UWE (ArgoUWE [5] for modelling, UWEXML for processing the models) allows the semi-automatic creation of websites. The tools promote the use of frames to subdivide pages into subpages – this is a feature which should not be overused due to the fact that frames are generally frowned upon by usability experts.

As evidenced by the thoughts on storyboarding, the authors of UWE have put some effort into creating a development method which ensures that the resulting websites have high usability. Nevertheless, the abstraction level of any "usability modelling" is comparable to that of OO-H: The available patterns are very concrete, and the presentation diagram is separate from the navigation diagram – a fact which may make it more difficult to create intuitive websites, as the two models are strongly related to each other, and changes to the one will rarely go without changes in the other.

4. CRITIQUE

The two Web Engineering solutions are similar in their approach, so any criticism regarding usability support is equally applicable to both.

The methods and tools show promise, but there is still room for improvement regarding the "usability of the development process". The use of the presentation models is not fully justified: They could be replaced with simpler template solutions without ill effects, primarily due to the fact that the existing user interface patterns supported by them are so straightforward that they can be expressed in simple template languages. Also, creating the presentation model results in work for the developer that does not pay off; HTML or XML page templates will typically *still* need to be created in addition to the model. Finally, the editors for the presentation model are not nearly as powerful as established web design software like Dreamweaver. They only support a subset of what modern web pages (using e.g. JavaScript) can do, and thus limit the developer in his possibilities.

With UWE and OO-H, there are separate models for navigation and presentation. As mentioned earlier, having just one model which includes both presentational and navigational aspects may be more desirable from a usability point of view, due to the fact that navigation and presentation are closely coupled: The one deals with the operation of the website at the page level, the other within a page. An example where this problem shows is OO-H's *Location Pattern*, which adds *navigation* information to pages even though it is a pattern that is used in the *presentation* diagram.

Regarding the navigation models, another issue worth noting is that the methods focus on links necessary for the application, not so much on "general" links like a site navigation menu, "related links" etc. The importance of intuitive general navigation structures should not be underestimated, because for many sites, users browse a site primarily because they search for information, and not because they want to actually use the site's central web application.

Neither UWE nor OO-H provides a way to model usability guidelines at an abstract level, e.g. "searching is the most important activity", "a purchase in the online shop should be possible in five minutes" etc. Whether such modelling is possible at all remains an open issue.

5. CONCLUSION AND FURTHER WORK

In this paper, UWE and OO-H were examined with regard to "usability issues", which can be subdivided into issues with the development process and issues with the generated websites. The following conclusions can be drawn from this work:

- Some of the usability aspects that arose during manual conversion of a model into a website cannot be expressed with the means of the analysed Web Engineering methods and tools.

- Existing Web Engineering tools address the conversion of models into a set of pages and do not have their primary focus on the usability of pages. Current research on problems with the presentation models is mostly restricted to device independence.

- The separation of the navigation and presentation models is problematic, since changes in one of the two models often affect the other one. It might be advantageous to unify the two models. At the same time, only the more abstract aspects of the presentation model should be retained (e.g. information about patterns employed on the pages), the concept of page templates should be moved from the presentation models to the "implementation" part of the method.

- Compared to classical Software Engineering, which usually stops modelling at the abstraction level of class diagrams, Web Engineering modelling currently takes place at a much greater level of detail. This makes the development process of current solutions work-intensive and complex for the developer.

Further work is necessary in a number of areas:

- The ideas in further methods and tools should be examined to see whether they solve some of the issues raised above. For example, WebML takes a different approach to the presentational aspects of modelling.

- It is necessary to find ways to express the more abstract usability issues in the models, preferably not just in UML comments, but in a way which allows tools to recognize and act upon this information.

- Related to the point above, it must be determined what "usability tool support" can look like. Such support is imaginable at various levels. When modelling at a more concrete level, the patterns (like *guided tour*) of existent tools are an example. At more abstract levels, the web developer's knowledge about typical usage patterns of a web application or about the expectations of users regarding website behaviour could be taken advantage of.

6. REFERENCES

[1] C. Cachero, J. Gómez, O. Pastor: Object-Oriented Conceptual Modeling of Web Application Interfaces: the OO-HMethod Abstract Presentation Model. In *Proceedings of the 1st International Conference on Electronic Commerce and Web Technologies EC-Web 2000*, London, UK, pages 206–215, Springer LNCS 1875, September 2000

[2] L. Constantine: Devilish Details: Best Practices in Web Design. In *Proceedings of the First International Conference on Usage-Centered, Task-Centered, and Performance-Centered Design forUSE 2002*, Rowley, MA: Ampersand Press, 2002.

[3] J. Gómez, C. Cachero, O. Pastor: Conceptual Modeling of Device-Independent Web Applications: Towards a Web Engineering Approach. In *IEEE MultiMedia* Volume 8, April–June 2001, pages 26–39

[4] R. Hennicker, N. Koch: Modeling the User Interface of Web Applications with UML. In *Practical UML-Based Rigorous Development Methods - Countering or Integrating the eXtremists*, Workshop of the pUML-Group at the UML 2001, A. Evans, R. France and A. Moreira, editors. Gesellschaft für Informatik, Köllen Druck+Verlag, pages 158–173, October 2001.

[5] A. Knapp, N. Koch, F. Moser, G. Zhang: ArgoUWE: A Case Tool for Web Applications. *First Int. Workshop on Engineering Methods to Support Information Systems Evolution* (EMSISE 2003), September 2003.

[6] N. Koch, A. Kraus: The Expressive Power of UML-based Web Engineering. Second International Workshop on Web-oriented Software Technology (IWWOST 2002), May 2002.

[7] J. Nielsen: Top Ten Mistakes in Web Design. Jakob Nielsen's Alertbox, `http://www.useit.com/alertbox/9605.html`, accessed Sep 1, 2004.

[8] J. Nielsen: When Bad Design Elements Become the Standard. Jakob Nielsen's Alertbox, November 14, 1999, `http://www.useit.com/alertbox/991114.html`, accessed Sep 1, 2004.

[9] B. Oestereich, C. Weiss, C. Schröder, T. Weilkiens, A. Lenhard: Objektorientierte Geschäftsprozessmodellierung mit der UML. dpunkt.verlag Heidelberg, 2003