Automatic Runtime Validation and Correction of the Navigational Design of Web Sites

Sven Casteleyn[1], Irene Garrigós[2] and Olga De Troyer[1]

[1] Vrije Universiteit Brussel, Department of Computer Science, WISE, Pleinlaan 2, 1050 Brussel, Belgium
{Sven.Casteleyn, Olga.DeTroyer}@vub.ac.be
[2] Universidad de Alicante, IWAD, Campus de San Vicente del Raspeig, Apartado 99 03080 Alicante, Spain
{igarrigos}@dlsi.ua.es

Abstract. Essential to an audience driven website design philosophy is the organization of information and functionality according to the requirements of the different audience classes. However, at design time, it is often difficult to correctly assess the different needs and requirements of the different prospective users of a website. This may result in a non-optimal navigation structure, which will decrease the usability of the website. In this paper, we describe how to correct, at run-time and automatically, possible flaws in the design resulting from incomplete requirement assessment, using adaptive behavior. By observing the browsing behavior of the users, the requirements for the different users are validated and the website is adapted according to adaptation specifications made by the designer. These specifications express when and how the website needs to be adapted and are expressed using an Adaptation Specification Language. The work is presented in the context of an audience driven design method but we also elaborate shortly on the applicability of the technique in general.

1 Introduction

In the beginning of the World Wide Web (WWW), when web sites consisted of one single page with hyperlinks, or a small amount of linked pages, the necessity for a well thought-out, structured design was not important: complexity was well manageable for web designers and the resulting site was easy to grasp for its visitors. However, as the WWW aged, and web sites became large, professional applications, offering both static and dynamic, rapidly changing information and functionality, maintenance and usability problems with ad hoc designed sites became apparent [16]. Visitors failed in making a mental model of the website, experienced difficulties in locating the information they were looking for and felt 'lost in hyperspace'.

Quite recently, we see that companies and organizations, in an attempt to better facilitate the user in finding information relevant for him/her, use a so called Audience Driven approach [6]. In this approach, the designer takes into account the target audiences of the web site, and creates the main structure of the site according to the information and functionality required by these different target audiences. Concretely, for the visitors this results in different links on the homepage, each representing a different navigation path (also called *audience track*) for a different type of visitor. Examples of leading companies and organizations applying an audience driven approach include[1]: HP (http://www.hp.com/), Sun Microsystems (http://www.sun.com/), IBM (http://www.ibm.com/us/), AIG (http://www.aig.com/GW2001/home/), NASA (http://www.nasa.gov/), Dell (http://www.dell.com/), several universities (Oxford University: http://www.ox.ac.uk/ and others),…

Although the audience driven design philosophy significantly reduces the amount of information the visitor needs to plough through, it also has as drawback that possibly some information needed is not available in the audience track chosen by the user. This is a direct result of the fact that the assessment of requirements by a designer (at design time) is reflected in the final navigation structure of the site, i.e. incorrectly assessed requirements result in information/functionality being in a wrong navigation track. More specifically, some relevant information may be missing in a certain audience track but present in another, while other information present in that audience track may be superfluous.

Indeed, for web designers it is very difficult, if not impossible, to correctly assess the exact requirements relevant or irrelevant for a certain user: it is often difficult or impossible to access the targeted web visitors and perform standard requirement engineering techniques. In this paper, we tackle this (requirement-) problem by describing how to identify missing or superfluous requirements for audience classes (section 4 and 5). Based on this identification, correction of the structure and navigation of the web site can be done automatically, at runtime (section 6). Note however that the specification of when, what and how to adapt is done at design time. This effectively gives the designer the means to anticipate possible requirement/design problems, and specify during the design how to correct for them if they are detected (at runtime). Preliminary results on this topic can be found in [3]. The work will be presented in the framework of the Web Site Design Method (WSDM), explained in section 3. Section 2 gives an overview of related work.

2 Related work

Using adaptation to better tailor a web site to the users exists in the adaptive hypermedia community (e.g. [18] and others). In this community, most of the work is done in the context of learning and user assistant applications. In these systems, the user is 'guided' towards a certain goal (mostly a *learning* goal), by (adaptively) constructing the pages tailored to his knowledge (which is e.g. gathered by analyzing browsing behavior). This approach is fundamentally different from the one taken in this paper: the navigation space is conditionally specified during design and the actual navigation space for a user is generated based on the profile of the user. The navigation space is in fact personalized. Our goal is different. We do not want to personalize the naviga-

[1] At the time of writing this paper

tion space; we only want to improve it using adaptive techniques. The adaptation is done after the (statically) designed navigation space has been used, and this usage data has been analyzed (note that the adaptation *is* specified during design).

Web design methods that have support for adaptation include WebML[4], Hera[10], UWE[14] and OOH[11]. However, these methods focus on personalization (e.g. content personalization) or/and customization towards different devices or context. They do not evaluate and alter an existing navigation space (after it has been deployed), as described in this paper.

WebML does describe a quality evaluation framework that exploits conceptual knowledge when analyzing web access logs [15]. The framework supports web usage analysis (e.g. access analysis of data schema entities at instance level) and web usage mining (e.g. finding data that is accessed together, navigation sequences). But there is no provision for (automatic) adaptation based upon this analysis.

3 WSDM overview

Given the fact that WSDM has been sufficiently specified in the literature [1] [2] [7] [8], we will only provide a short overview and go into deeper detail only where necessary for the context of this paper.

The general design cycle for WSDM can be summarized as follows. In the mission statement, the purpose and the subject of the web site is expressed and the target audience is identified. Based on the mission statement, an audience modeling phase is performed, consisting of an audience classification and an audience characterization. During audience classification, the different audience classes based upon the informational and functional requirements of the potential visitors are identified and classified into an audience class hierarchy. During audience characterization, the relevant characteristics for the different audience classes are described.

The next phase, the Conceptual Design, consists of two sub-phases: task modeling and navigation design. During task modeling, for each requirement of the different audience classes, a task is defined. Consequently, the task is elaborated and decomposed in elementary tasks using ConcurTaskTree ([7] describes how). In this way, a task model is specified for each requirement. In parallel, for each elementary requirement, a tiny conceptual schema, called a chunk, is specified, formally describing the information or functionality needed to fulfill the requirement. These chunks are modeled using ORM [12] with some extensions to express functionality.

During navigational design, the conceptual navigation model for the website is constructed. It consists of a graph of nodes, connected by means of links. Every node contains information that logically belongs together, in the form of (information or functional) chunks. A node can contain zero or more chunks (a node without chunks represents a kind of *hub*, presenting a choice of navigation possibilities to the user). In WSDM, the navigation model is obtained in two steps. First, the audience class hierarchy from the audience modeling phase is used to obtain the basic structure of the navigational model, by creating one navigation track (a so called *audience track)* for every audience class [1]. Such an audience track can be thought of as a “sub-site”, where all and only the information/functionality relevant for that particular audience class can be found. Secondly, the internal structure of each audience track is derived from the task models of the task modeling phase, elaborating the requirements of each audience track.

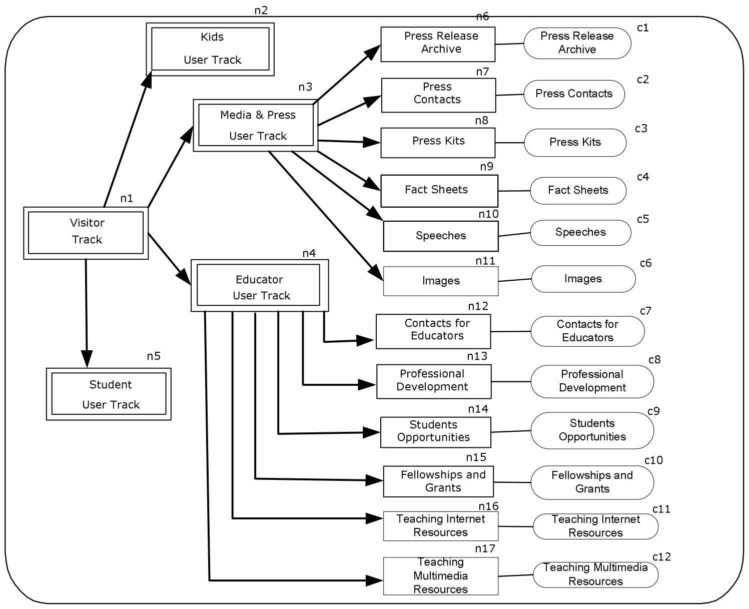

Figure 1. Simplified Navigational Model for (part of) the NASA website

It is only in the next phase, the implementation design, that the conceptual nodes from the navigational model are actually mapped on pages, and that layout and presentation is specified. Finally, the last phase, the actual implementation, is a mapping of the previous phases onto the chosen implementation platform.

A (simplified) example of a Navigational Model of an audience driven website is given in figure 1. The example is the NASA website (http://www.nasa.gov/) and shows the main structure of the website and the navigation structure for both the Media & Press User Track and the Educator User Track. The other tracks are not elaborated to not overload the figure. Nodes are represented by rectangles, chunks by rounded rectangles, links by arrows and the connections between chunks and nodes by lines. A double lined rectangle is used for the root of a track or the root of the web site (in this example the node ‘Visitor Track’ is the root of the web site).

4 Missing or Superfluous Information

As already explained in the introduction, the aim in this paper is 1) to detect missing or superfluous requirements for a certain audience class by investigating the access to information offered in the corresponding audience track, and 2) to adaptively alter the structure of the site to correct for detected problems. In this section, we will describe the data necessary to be able to identify missing or superfluous requirements. In the next section, we will describe how to use this data to detect flaws, and in section 6, we will give possible solutions to correct the flaw.

From now on, the requirements originally assigned (by the designer) to an audience class will be called *native* requirements; all other requirements will be called *foreign* requirements (for that particular audience class).
According to the audience driven philosophy, a user should find *all* the information and functionality he needs in his particular audience track. Consequently, if a user leaves his audience track, to look for information elsewhere, then the information he is visiting outside of his audience track might be information that is actually missing in his own audience track. If a significant amount of users of the same audience class do this, then we have a good indication that that information is actually relevant for this audience class.

When a user does not leave his audience track, still some problems with the information in the track are possible. Some information in the track might be accessed very few times (compared to the other information), suggesting the information does not belong there in the first place. Note that identifying information that is accessed few times does not necessarily imply that information is superfluous. We will discuss this in more detail in section 5.

Thus, to identify missing/superfluous information, the following steps are identified:

1. Determine the audience class of the current user
2. Track which information the user visits, both in and outside his audience track
3. Analyze the accumulated data to determine if the information within the audience track is relevant, or if visits to information/functionality outside the audience track are significant

Because an audience driven design has been used, the first step is simple: when a user enters the web site, he is presented with a choice of different audience tracks (i.e. subsites), each representing the information/functionality for a certain type of user. By selecting one of these tracks, the current user effectively identifies the role in which he is currently visiting the website, and thus reveals his/her audience class.

In the second step, we store data about which information a user visits. As we want to keep track of which information is visited outside a particular audience track, and relate this information to the frequency of visits inside the track, we need to store the number of visits to each piece of information (chunk) relative to each audience class. This data can be conveniently stored in a matrix, which we will call the *information access matrix*. Rows of the matrix represent the different elementary requirements R_i

(track-able in the website by their corresponding chunk C_i), while the columns represent the different audience classes. An example is given in the next section.

In pseudo code, the algorithm to populate the matrix is as follows:

```
WHEN session.start THEN Determine Audience Class $A_j$ by observing first click(s)
    WHILE browsing DO: IF currentNodeVisited has chunk(s) $C_i$ connected
                       THEN FOREACH ($R_i$) : M(i,j)++
    END
END
```

Note how the connection between the requirements (line 3) and their corresponding chunks (line 2) is exploited. Over time[2], the matrix contains a good summary of the amount of accesses to the different chunks for each audience class. In the next section, we will give an example of an information access matrix, and we will describe how the matrix can be used to identify missing or superfluous information in an audience track.

5 Method for Identifying Missing or Superfluous Information

5.1 Identifying Missing Information

To determine if the amount of accesses from a certain audience class to information outside the audience track is significant, we can use known statistical techniques. In statistics, the problem of determining if a certain value *fits well* in a given relatively small sample data set is solved using basic statistical measures for central tendency (e.g. mean, median, ..) and standard measures of spread (e.g. variance, standard deviation, median absolute deviation, ..).

As our dataset (and its distribution) is unpredictable and we do not want our calculations influenced by (a few) extreme values (e.g. nodes that are highly popular), we choose median, a robust central tendency (i.e. not influenced by extremes in our data set). As a measure of spread the median absolute deviation (MAD) is chosen, as this measure is less influenced by outliers (compared to other measures) and has robustness of validity (i.e. the underlying distribution doesn't influence reliability too much). The following formula is used to calculate absolute median deviation:

– $MAD = median\ (|x_i - x_m|)$

where x_i is each of the values of the data set, and x_m is the median of the data set.

[2] An exact estimate the (first) evaluation time of the matrix is not the subject of this paper; suffices to say a significant amount of visitors needs to have visited the site.

As the spread denotes how far, on average, the values of the dataset are removed from the middle value, we can conclude that most of the values of the dataset lie within the distance of the spread from the middle value (for more exact estimates of how much elements of the data set lie within this interval, we refer to [5]). Consequently, external values *fit well* in the given dataset, if they lie within that range.

Applied to the problem of finding information outside the audience track of a particular audience class that is actually relevant for that particular audience class, we adopt the following method:

For a given audience class/track:

1. Calculate median and MAD for the set of number of accesses to information resulted from *native* requirements and calculate the threshold (median – MAD)

2. For all information resulting from *foreign* requirements, verify if the amount of accesses is greater than the calculated threshold. If this is the case, we have an indication that that particular information is relevant to the audience class under investigation

Note that only the lower limit of the range is used as information resulting from a foreign requirement track that is accessed more than (median + MAD) is off course also relevant for the current audience class.

To clarify this method, let's consider the (simplified) example of the NASA web site that was introduced in section 3 (see figure 1). The information access matrix for this example is shown in figure 2. In the columns we have the two audience classes considered in this example (Media & Press and Educators) and in the rows we have the different requirements that resulted in information chunks on this web site. As explained, each cell in the matrix denotes the number of visits to some information by a certain audience class. For example, cell (1,1) shows that members of the audience track "Media & Press" have accessed 52 times the information resulting from requirement R1 "*Press Release Archive*".

Note that the first six requirements in the matrix are native for the *Media & Press* track, and the last six ones are native to the *Educator* track. As we were unable to obtain the *real* access information for the NASA website, we have used here fictitious data.[3]

Lets now analyze the accesses to the native information of the Educators audience track (R7 … R12), and determine if accesses to foreign information (R1 .. R6) were significant compared to these accesses. Calculating median and MAD we obtain:

Data set (ordered): 10 15 20 30 50 56 ; **Median:** 25 ; **MAD:** 12.5

[3] Experiments are being performed at the author's university web site with real access data.

		Media & Press	Educators
R1	Press Release Archive	**52**	**40**
R2	Press Contacts	**49**	**4**
R3	Press Kits	**31**	**10**
R4	Fact Sheets	**16**	**5**
R5	Speeches	**40**	**5**
R6	Images	**38**	**12**
R7	Contacts for educators	**0**	**56**
R8	Professional development	**5**	**50**
R9	Student opportunities	**1**	**30**
R10	Fellowships and grants	**0**	**10**
R11	Teaching Internet Resources	**3**	**20**
R12	Teaching Multimedia Resources	**2**	**15**

Figure 2. Information Access Matrix

The threshold that is the lower limit to decide if foreign information is relevant for the current audience track is median – MAD = 25 – 12.5 = 12.5. The information of the *Media & Press* track with a number of accesses greater then this threshold is determined to be relevant for the *Educator* audience track. This is the case for the *Press Release Archive* information (40 accesses). We can thus conclude that this information should somehow be included in the Media & Press track (how and where this information is added is the subject of the next section).

5.2 Identifying Superfluous Information

As for identifying missing information, we also use statistical techniques to determine when information is superfluous. In statistics, the problem of finding an outlier in a data set is generally seen as hard, especially when the dataset is small, and accurate information about it (e.g. distribution) is missing. Consequently, most existing tests are not usable here because they only work on large datasets (e.g. Rosner's Test [17] and others [5], [13]), and those that were usable for small data sets gave poor results (e.g. Dixon's test [9] and others).

However, for our purposes, detecting *significantly low* values (but not per se outliers as they are defined in statistics) in our dataset is already sufficient. To identify these *low* values, we will use a double strategy: we will look for values which lay both *far* from their (bigger) neighbor, and also lay far from the middle value of the dataset.

To determine which value lies far from its neighbor, we take the ordered dataset, and calculate the distances between each 2 consecutive values. These distances give us a new dataset, for which we calculate mean and standard deviation[4] (= std). Calculating the interval [(mean–std) (mean+std)] gives us the range in which most of the distances (i.e. 50% for normally distributed data) lie. Distances above the (mean+std)

[4] As this time, we want to detect high distances, we use mean and standard deviation, as they are more affected by the presence of extreme low or high values.

are thus relatively big, compared to the other distances, and we have identified two points which are relatively far from each other.

To determine which point lies *far* from the middle value, we apply the same technique as in the previous section: values below the threshold (median–MAD) can be labeled as being *far* from the middle value.
Let's consider the NASA website example from the previous subsection, and apply the technique described above to identify possible superfluous information in the Media & Press track:

Data set (ordered): 16 31 38 40 49 52	Data set of distances: 15 7 2 9 3
Median: 39 ; MAD: 9	Mean: 7.2 ; Std.: 4.7
Lower limit: 39 – 9 = 30	Upper limit: 7.2 + 4.7 = 11.9

The distance between the first and the second element of the original dataset is 15, which is bigger than the threshold 11.9. Therefore, we can conclude that the first element lies significantly *far* from the next element. As this first element, namely 16 in the original dataset, lies below the threshold of 30, we can also conclude that it lies *far* from the middle. With both tests positive, we conclude that the value 16 is indeed significantly low compared to the other values: the information related to the requirement 'Fact Sheets' is detected as (possibly) superfluous for the audience class Media & Press. The concentrated reader will note that, in case the detected value is not the first value of the data set (as in this example), then also all smaller values are marked as superfluous, as they obviously are also significantly low compared to other values.

6 Adaptation of the Web Site Structure

Having identified missing or superfluous information in a certain audience track, the structure of the web site can be adapted (automatically) to reflect the detected deficiencies. To leave the control over the web site structure to the designer, we allow the designer to specify at design time how the structure of the web site should be adapted when such deficiencies are detected. For this, the Adaptation Specification Language (ASL) [2] can be used, which is defined upon the navigational model to allow specifying (at design time) possible adaptive changes to the structure of the site (at run time). Due to space restrictions, we have simplified ASL notation in this paper to reflect the basic idea.

6.1 Adaptively correcting missing information

There are several ways to anticipate the fact that information present in some audience track is apparently missing in a certain audience track. The designer might duplicate the information in the place it is missing; provide a link to the existing information from where it is missing; or totally re-arrange the structure of the site so that all interested users can access the information.

The approach taken in this paper consists of "duplicating" the information, and offering a link to the information at the root of the audience track. Although more suitable adaptations can be chosen, we think that the ratio effort/benefit is certainly the highest for this adaptation.

Let W be a web site, with a set of Audience Classes A={A_1, ..., A_n} with associated Audience Tracks T={T_{A1}, ..., T_{An}} and MR_{Ai} the set of nodes containing information detected to be missing from the audience track T_{Ai}. We can express the adaptation explained above as follows:

```
(1)  Foreach AudienceClass in A
(2)    Foreach node not in NodesOfAudienceTrack(AudienceClass):
(3)      if node in $MR_{AudienceClass}$ then addLink(root($T_{AudienceClass}$), duplicate(node))
```

In words, this rule iterates over all audience classes (line 1). Within each audience class, for each node outside the audience track of that audience class (line 2) it is checked if that node contains *missing* information (line 3). If this is the case, the node is duplicated (along with all chunks connected to it) and a link from the root of the audience class under investigation to the duplicated node is added (then-part line 3).

6.2 Adaptively correcting superfluous information

As mentioned in section 4, information identified as superfluous in a certain audience track does not necessarily need to be removed. Although visited only few times, it might still be valuable for (a small amount of) visitors (think of 'how to get there' information for a research lab: only a minority of users will actually want to go to the lab, yet the information is invaluable on the website). Thus, we see the detection of superfluous information rather as an alert to the web master, rather than something that requires (automatic) adaptation. If the web master indeed decides the information is not needed in that track, he can remove it. However, an (automatic) adaptation that may be useful for the designer to specify, is re-arranging the links so that a link to information detected as superfluous appears last.

```
(1)  Foreach AudienceClass in A:
(2)    Foreach node in NodesOfAudienceTrack(AudienceClass):
(3)      if node in $SR_{AudienceClass}$ then
(4)        foreach link with target(link) = node: reorder(link, last)
```

In words, the rule iterates over all audience classes (line 1). For each node within the audience class (line 2), it is checked if the node has been detected to contain superfluous information (line 3). If this is the case, then each link to this node (line 4) will be re-ordered so it appears last in the originating node (end line 4).

7 Applicability beyond the Audience Driven Approach

The technique, described in this paper, to validate requirements by detecting missing or superfluous information in a certain audience track, can also be applied to web sites not following the audience driven philosophy. The key requirement for the applicability of the approach is the existence of a collection of information belonging logically together (in our case, the requirements for an audience class). Information in such a collection can be tested to be superfluous, and information external (to this collection) can be tested to be missing. Two example applications are given below.

In a data driven web site, where the navigation structure is determined by the available data, the main navigation structure (e.g. the main navigation options from the homepage) is usually topical. The technique described in this paper could be used to determine relevant data outside a certain topic group, and possibly adaptively alter the navigation structure to cluster related topic groups or to provide a link within the topic group to the (external) relevant data.

In e-commerce, product categories can be considered as sets of related items, and thus the technique described in this paper could be used to determine which products from other categories could be of interest to a user browsing a particular category. E.g. if a user is leaving a product category, briefly browses outside the category, and consequently returns, this may indicate that the information visited outside the category may also be relevant for the current category. Consequently, the navigation structure could be adaptively altered to include a link to the categories in question.

8 Conclusion

In this paper, in the context of WSDM, we show how to validate the requirements of the different users by detecting two types of flaws in an audience driven web site. These flaws are 1) information put wrongly in a certain track (superfluous information), and 2) information missing in a certain track but present in another (missing information). The automatic detection is done using statistical techniques, performed on the web access log. Furthermore, this paper describes how the existing navigation structure can be (automatically) adapted to remedy for detected flaws: at design time, the designer can specify (using the Adaptive Specification Language) the adaptive actions that should be taken in case such situations are detected at run time. By doing so, the structure of the web site will better reflect the audience driven design philosophy, and the site will be better tailored to the needs of the different audience classes. Finally, the paper elaborates on the use of this technique outside the scope of audience driven web design.

References

1. Casteleyn, S., De Troyer, O.: Structuring Web Sites Using Audience Class Hierarchies, In Proceedings of DASWIS 2001 workshop (ER conference), Yokohama, Japan (2001)
2. Casteleyn, S., De Troyer, O., Brockmans, S.: Design Time Support for Adaptive Behaviour in Web Sites, In Proceedings of the 18th ACM Symposium on Applied Computing, Melbourne, USA (2003)
3. Casteleyn, S., Garrigós, I., De Troyer, O.: Using Adaptive Techniques to Validate and Correct an Audience Driven Design of Web Sites, In Proceedings of ICWE'04 (2004)
4. Ceri S., Fraternali P., and Bongio A: Web Modeling Language (WebML): a modeling language for designing Web sites, In WWW9 Conference, First ICSE Workshop on Web Engineering, International Conference on Software Engineering (2000)
5. DeGroot, M..H.: Probability and Statistics. Addison-Wesley, Reading, Massachusetts (1989)
6. De Troyer, O.: Audience-driven web design, In Information modelling in the new millennium, Ed. Matt Rossi & Keng Siau, IDEA GroupPublishing, ISBN 1-878289-77-2 (2001)
7. De Troyer, O. ,Casteleyn, S.: Modeling Complex Processes for Web Applications using WSDM, In Proceedings of the IWWOST2003 workshop, Oviedo, Spain (2003)
8. De Troyer, O. ,Leune, C.: WSDM: A User-Centered Design Method for Web Sites, In Computer Networks and ISDN systems, Proceedings of the 7th International World Wide Web Conference, Elsevier (1998) 85 – 94
9. Dixon, W. J.: Analysis of extreme values. Ann Math Statist, 21 (1950) 488 - 506
10. Frasincar, F., Houben G., and Vdovjak R.: Specification Framework for Engineering Adaptive Web Applications, In Proceedings of the WWW Conference, Honolulu, USA (2002)
11. Garrigós, I., Gómez, J. and Cachero, C.: Modelling Dynamic Personalization in Web Applications, In Proceedings of ICWE'03, LNCS 2722 (2003) 472 - 475
12. Halpin, T.: Information Modeling and Relational Databases: From Conceptual Analysis to Logical Design, 1 st Edition. Morgan Kaufmann Publishers, San Francisco, USA (2001)
13. Hawkins D.: Identification of Outliers, Chapman and Hall (1980)
14. Koch, N.: Software Engineering for Adaptive Hypermedia Systems, Reference Model, Modeling Techniques and Development Process, PhD Thesis, Verlag UNI-DRUCK, ISBN 3-87821-318-2 (2001)
15. Lanzi, P.L., Matera, M., Maurino, A.: A Framework for Exploiting Conceptual Modeling in the Evaluation of Web Application Quality. In Proceedings of ICWE2004, Munich (2004)
16. Nielsen, J.: Finding usability problems through heuristic evaluation. Proceedings of the SIGCHI conference on Human Factors and Computer Systems. Monterey, California, United States ISBN:0-89791-513-5 (1992) 373 – 380
17. Rosner, B.: On the detection of many outliers. Technometrics, 17, pp. 221-227.(1975)
18. Wu., H., Houben, G.J., De Bra, P.: AHAM: A Reference Model to Support Adaptive Hypermedia Authoring, In Proc. of the Zesde Interdisciplinaire Conferentie Informatiewetenschap, Antwerp (1998) 77 - 88