# Weighting Task Procedure for Zoomable Task Hierarchy Modeling of Rich Internet Applications

Francisco J. Martínez-Ruiz[1], Jean Vanderdonckt[1], and Jaime Muñoz[2]

[1] Université catholique de Louvain, Belgian Lab. of Computer-Human Interaction
Place des Doyens, 1 – B-1348 Louvain-la-Neuve, Belgium
{francisco.martinez,jean.vanderdonckt}@uclouvain.be
[2] Universidad Autónoma de Aguascalientes, México
jmunozar@correo.uaa.mx

**Abstract.** Zoomable user interfaces are more attractive because they offer the possibility to present information and to support actions according to a "focus+context" method: while a context of use is preserved or presented in a more compact way, the focus can be achieved on some part of the information and actions, enabling the end user to focus on one part at a time. While this interaction technique can be straightforwardly applied for manipulating objects of the same type (e.g., cells in a spreadsheet or appointments in a calendar), it is less obvious how to present interactive tasks of an information system where tasks may involve very different amount and types of information and actions. For this purpose, this paper introduces a metric based on a task model in order to decide what portion of a task model should lead to a particular user interface container, group, or fragment, while preserving the task structure. Each branch of the task model is assigned to a weight that will lead to such a container, group, or fragment depending on parameters computed on variables belonging to the context of use. In this way, not only the task structure is preserved, but also the decomposition of the user interface into elements depends on the context of use, particularly the constraints imposed by the computing platform.

**Keywords:** Multiple contexts of use, Rich Internet Applications, Task-based metric, Task Tree modeling, Task weighting, Task coding schemes.

## 1 Introduction

The task model has been widely recognized as a rich source for initiating the development life cycle of a User Interface (UI) of an interactive system [27]. When it comes to designing a UI for multiple contexts of use, for instance on different computing platforms, the task model has also been exploited in order to drive the process of deciding how the UI will be decomposed into screens and how the transition between these screens will be ensured. This paper tackles a specific necessity of the first step in the method proposed in [5] which was conceived as a Model Driven approach for the developing Rich Internet Applications (RIAs): How to calculate (and justify) the weight of task Hierarchies?

D. England et al. (Eds.): TAMODIA 2009, LNCS 5963, pp. 92–102, 2010.

This initial task is the specification of user goals as a task hierarchy, i.e., a task hierarchy model of the application (THM). The procedure is indeed an iterative process, beginning with general tasks which are decomposed into simpler tasks. This work produces structures of variable size (for non-trivial developments the caliber increase very quick). The specification in the previous version of the method is based on ConcurTaskTree notation [15]. One of the drawbacks of this notation is the visual ambiguity. That is, the repetitive structures which conforms them are not prepared to help in the process of pattern recovery or semantic inferences. In order to resolve this problematic, The Zoomable User Interface (ZUIT) is used. For instance, this alternative visualization can help developers to: (1) Finding hidden patterns and (2) Identifying unbalanced hierarchies (i.e., putting many tasks in a sub tree) which intuitively implies a poor UI design.

The rest of this paper is organized as follows: Section 2 discuss the state of the art in the creation of ZUIs. Section 3 introduces the reference framework. Then Section 4 covers the description of our method over a minimalistic case study. In Section 5, another example is presented. And finally, Section 6 presents conclusions and future work.

## 2 Related Work and Problem Description

This section includes three subsections: first, a brief review of the state of the art in the domain of RIAs. Second, one dedicated to Zoomable User Interfaces. And third, an analysis of the problem in terms of weight measure.

### 2.1 RIA Complexity

The complexity of RIA applications involves multiple factors. First, there are a lot of elements to coordinate in order to model the UI presentation. Second, they include unusual widgets with nontraditional behavior. Third, the web page metaphor is not maintained. That is, reloading of web pages is substituted by a continuously present interface with soft transitions [5], [17].

### 2.2 Review of Zoomable User Interfaces

The selection of Zoomable user interfaces (ZUI) is based on the benefits that could afford to RIA development. For instance, searching information in schematic diagrams [10]. ZUIs are suitable for dealing with hierarchical structures (e.g., images or 3D scenarios) and vast sets of information [3]. In [9] 2D structures were used in order to display the internal disposition of files in a directory structure of hard disks. Note: Here instead of dealing with files of variable size we have tasks with different complexities. Also, in [10] one of the contributions of ZUIs is the sense of location combined with object constancy (i.e., I could browse the THM without losing resolution and sense of location). In [7] and [14] the advantages of ZUIs in comparison to window based ones are studied. It is possible to find interesting examples: a calendar application for mobile devices that combines fisheye views with reduced overviews [12]. Another example is a time-line with zooming facilities [13]. The development of this kind of applications does not follow a model driven approach.

### 2.3 Problem Description

In [5] the first step of development includes THMs as the initial phase. But the used THMs are weightless structures and there is a lack of sense of complexity. RIAs are very complex and the developer needs support in the moment of designing the definition of the application in terms of THMs. Therefore, the objective of this research is proposing a weighting process in order to produce more balanced structures. The sub goals of this include (A) a weighting function that takes into account the structure (locally and globally). (B) The structure could aid to discover easily overloaded sections of the application. (C) Providing a plausible alternative to current THM solutions in terms of efficient space utilization and interactivity (thanks to the ZUI approach). Finally, it is worth to mention that building task models [15] is not a trivial task. Even with the inclusion of limited zooming of some tree branches, word wrapping and fisheye treatment [23], the visualization (e.g., labels) and complexity description problems remain unresolved.

## 3 Alignment with the CAMALEON Reference Framework

In order to give a self contained presentation of the method, in the following section is briefly reviewed and after that, the focus of this paper is retaken. The method proposed in [5] uses as building scheme, the CAMELEON framework [24] and includes as first step the definition of a task hierarchy model or task tree (THM) in order to describe User goals (this level is called Task and Domain level). This model is then transformed into an abstract definition, called AUI and then to a concrete one, called CUI. Here, modality and type of widget selection are completed. Then, the last transformation is done. This Final UI is built in a specific platform (LZX, .NET, SWF among others). Note: it is a Model Driven compliant method as defined by OMG [16].

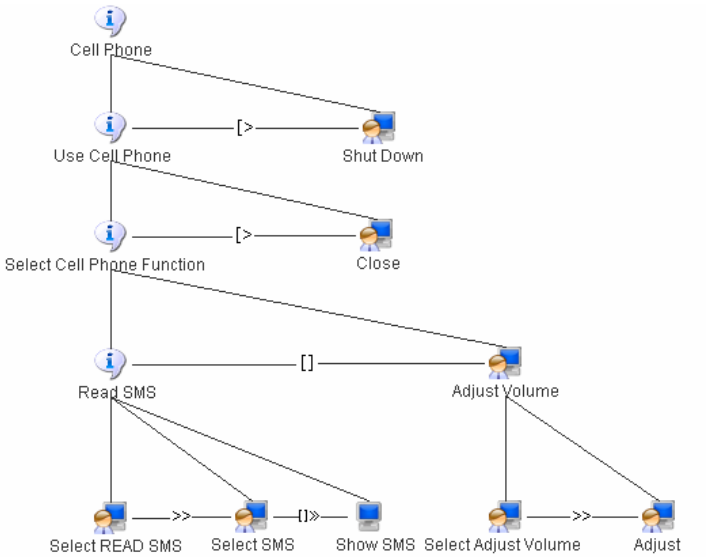


**Fig. 1.** Example of THM of some functionalities of a cell phone

### 3.1 Generating the Alternative Representation of Task Model

ZUIT representation of THMs uses a constant display area in order to show THMs of any size. The THM is traversed in a breadth-first order (see Fig. 3) and the transformation process is completed by XSL templates. The ZUIT definition includes two steps. (1) The weight-less ZUIT is first created (see Fig. 2). Here the inner boxes are spread in a regular order. That is, they take all available space like HTML table cells. And (2) each cell is updated in order to represent in a better way the complexity of the application (see section 4). In Fig. 1, we could see an example of a THM for describing some functions of a cell phone (adapted from [27]).

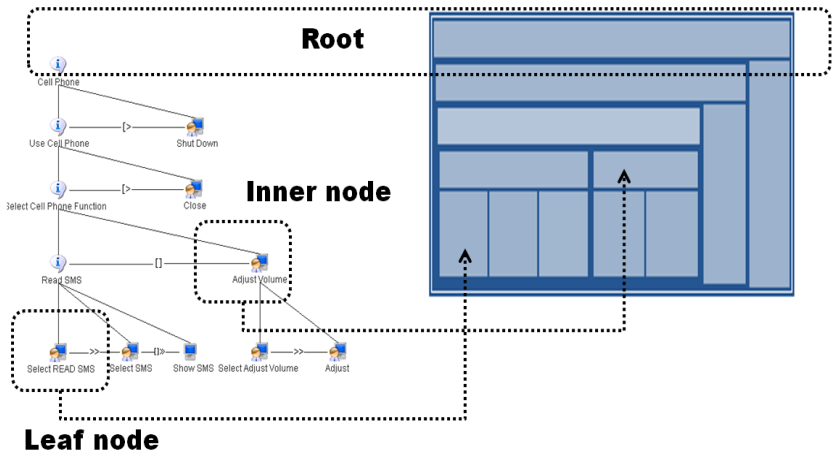


**Fig. 2.** Transformation of THM into ZUIT format

```
Function CreateZUIT structure(CTT tree)
returns ZuiStructure or failure
initialize the search tree to root node
loop do
     if there are no candidate nodes for expansion
     then return exit
     choose a node and expand its sons
            if node type equals parent-node then
                  GeneratBoxComponent()
            if node type equals leaf-node then
                  GeneratCellComponent()
                  DefineContainmentRelation(leaf, parent)
```

**Fig. 3.** A possible algorithm for generating the ZUIT structure

## 4 Method Outline

The following section describes the proposed refinements in order to calculate the weight (see Figure 4). A first step is the replacement of textual definitions by a color

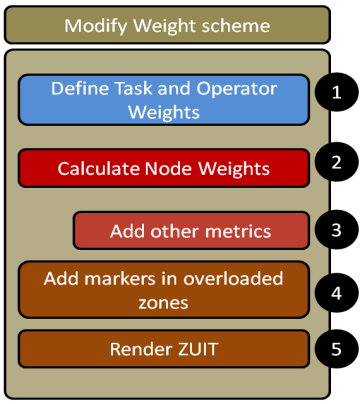


**Fig. 4.** The process of updating the ZUIT

**Table 1.** Coding scheme

| Color | Operator/task Type | Operators |
|---|---|---|
| | Sequential | [>, |>, >>, []>> |
| | Concurrent | |=|, ||| , |[]| |
| | Choice | [] |
| | Interactive task | |
| | System task | |
| | Abstract task | |

coding scheme in order to deliver visually simpler THMs (since it is not practical to deal with hundreds of labels). Also, the temporal operators are coded in colored headers which meaning could be seen in table 1.

The application of this process could be seen in Fig. 5. This process is done with the purpose of reducing the cognitive overload of the developer. Note: in order to make clearer the explanation, task names are included in Fig. 5 but in the real ZUIT task labels are included only as tool tips.

### 4.1 Definition of Task and Temporal Operator Costs

The complexity of each task is expressed with a specific weight (or cost). That is, the weight of each sub THM is directly proportional to its complexity. The cost of each type is defined with an exponential growth function, $\mathbf{y = 2^n}$ where $\boldsymbol{n}$ is a value between one and three. The rationale behind this selection includes two empirical reasons: (1) the minimal THM involves two tasks which give us the Base and (2) Sequential temporal operators involve one active task at a given time, so they have the lower cost. A choice operator before the selection involves more complexity but after

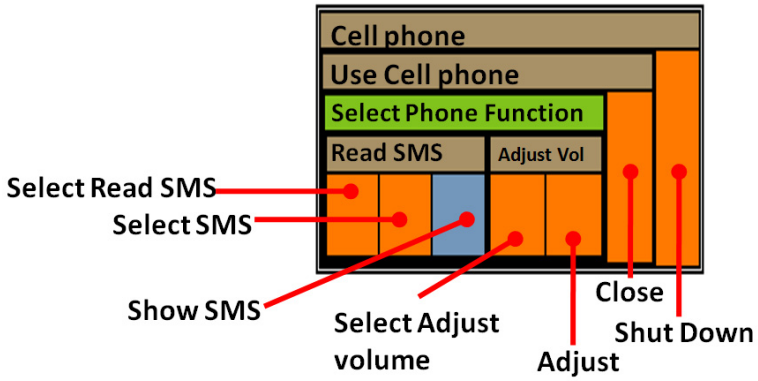


**Fig. 5.** ZUIT representation of some cell phone operations

the selection only one way is taken. Finally, concurrent operators imply execution in parallel, and then their cost is the biggest. Note: These weights were proposed in [17]. Meanwhile, the task types are also based in terms of the same logic: an application task is simpler (always from the UI perspective since is an output process in most of the cases) than an interactive task which implies a dialogue. Then, the second doubles the first in terms of cost (see table 2).

**Table 2.** Weight of THM elements

| Weight | Task Type | Operator type |
|---|---|---|
| 8 | - | Concurrent |
| 4 | Interactive | Choice |
| 2 | Application | Sequence |

### 4.2 Calculate Node Weights

The following function (1) is proposed in order to produce node weights.

$$W = \frac{n}{2}\left(\frac{\sum_{i=1}^{n} T_i \times TaskWeight + \sum_{j=1}^{m} O_j \times OpWeight}{deep}\right) \quad (1)$$

Where, **n** is the number of tasks in the sub tree, the division by two is related with the size of the minimal possible tree, i.e. the weight is understood as proportional increments of the weight of the minimal tree. **T** is the set of task nodes and/or head nodes of sub trees that are part of the current sub tree. **O** is the set of operators in the current level. Note: m= n-1, also for the sake of simplicity, we assume only a single temporal operator type in the case of different types, dummy tasks are created to cover and send them down. **Deep** is the distance from the root to the set of nodes to be weighted, the rationale behind the introduction of this value is to reduce the weight of sub trees

**Table 3.** Weight of ZUIT elements

| Abstract Container | Weight |
|---|---|
| Read SMS | 5,25 |
| Adjust Volume | 2,5 |
| Select phone function | 7,75 |
| Use cell phone | 6,875 |
| Cell phone | 12,875 |

formed by leave nodes or in the lower level of the task hierarchy, since they are related to simple tasks (maybe directly translated to specific widgets). The result of applying this to the example depicted in Fig. 2 could be seen in table 3.

### 4.3 Add Other Metrics

Other metrics could be added in order to refine the weighting process. For instance, centrality (CE) is a plausible one since introduce a notion of weight in terms of local and global position of tasks [25]. This metric takes into account the connectedness of a node. The rationale behind this metric is that inner nodes ranked with a considerable centrality value are the head of complex sub hierarchies.

### 4.4 Add Markers in Overloaded Zones

Another automatic process to be included is the control of overloaded areas. For instance, in Fig. 6a the number of children tasks is more than seven (indeed, there are fourteen tasks) and according to [26] this is not suitable (this value is used here in order to introduce a known capacity value). Then, the header's color is changed to red (see Fig. 6b). And there is a possible mechanical solution of subdividing the task into subtasks up to a specific capacity value. In Fig. 6c, the capacity is established to seven and Fig. 6d takes remaining tasks. Also more complicated algorithms could be introduced. For instance the one proposed in [28] but it requires adding more info about tasks and their translation to specific widgets and contexts.

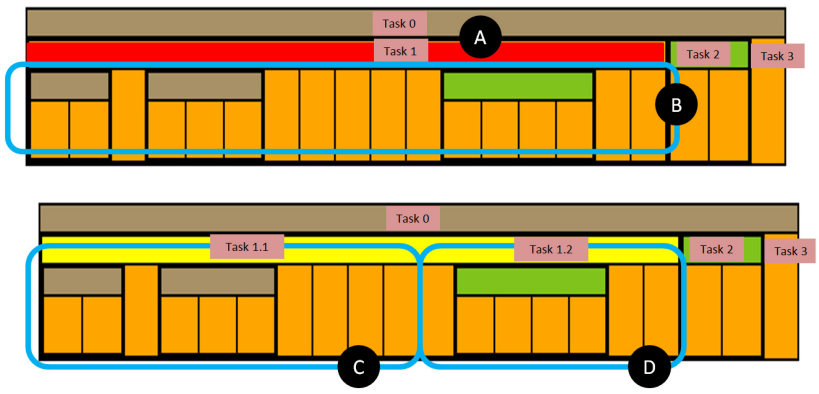


**Fig. 6.** Zooming process of the ZUIT for using a cell phone

### 4.5 Render the ZUIT

The process of rendering is done with the PICCOLO library [19]. The hierarchy of nodes is rendered into 2D graphical elements with the proper area and it could be navigated through zooming in and out operations (see Fig. 7).

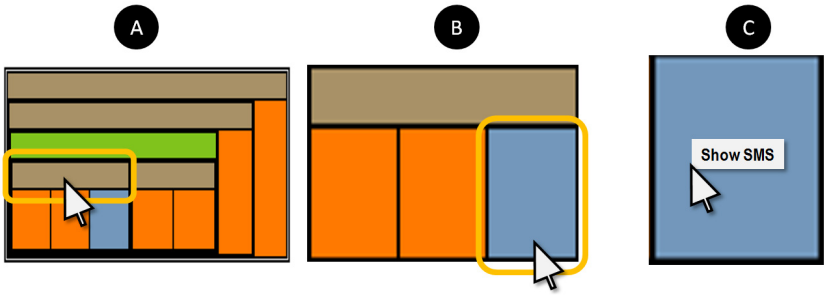


**Fig. 7.** Zooming process of the ZUIT for using a cell phone

## 5 Other Example: Cocktail Selection Application

In this section another example is presented. This application is used to select cocktails (see Fig. 8) also combines Zoomable capabilities with the expression of the ZUIs (here in gray scale in Fig. 9). Also we show here a possible THM (see Fig. 10).

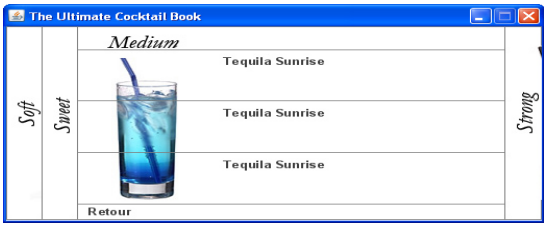


**Fig. 8.** Cocktail selection application

**Fig. 9.** Cocktail ZUIT representation

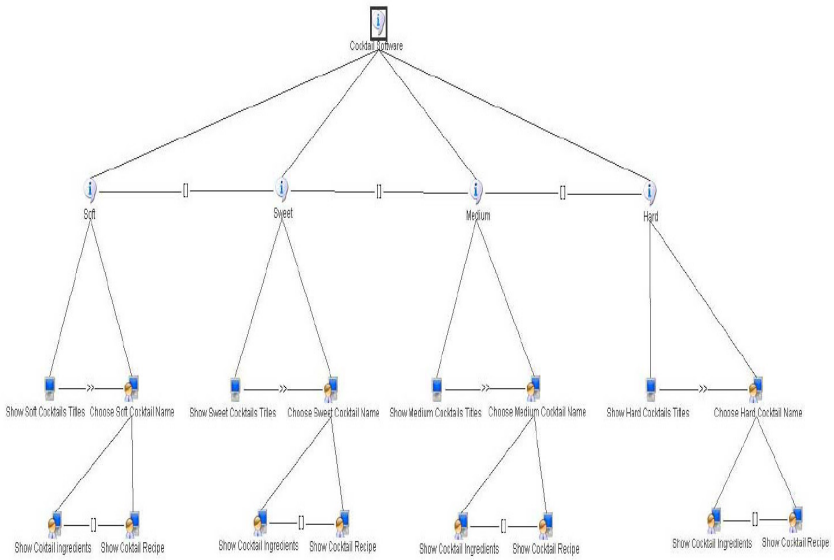


**Fig. 10.** Cocktail typical THM representation

## 6 Conclusions and Directions for Future Research

In this paper we introduce an updated weighing procedure (with the proposition of a weighting formula) in order to deal with the complexity of different task hierarchies in the development of RIAs. This weighting process is applied in Zoomable User Interfaces in order to surpass some of the problems of current THM approach. Piccolo framework [19] is used to implement the ZUITs. Finally, this approach is not only applicable to RIAs and it could be used for others types of applications.

### 6.1 Future Work

Only a couple of metrics were used in this first version of the function but we are introducing more metrics as these would continue to move our approach to its goal of being a known alternative representation of task models. In the meantime, a piccolo prototype is under construction in order to test the feasibility and adding more capabilities. Finally, more semantic exploration and elements could be added to our representation e.g., usability weights in terms of color ranges (from light to dark in order to represent adherence to usability guidelines).

## Acknowledgments

This work is supported by the Programme AlBan, the European Union Programme of High Level Scholarships for Latin America, scholarship No. (E06D101371MX) and the Belgian Computer Human Interaction Lab.

## References


1. Dachselt, R., Frisch, M.: Mambo: a facet-based zoomable music browser. In: Proceedings of the 6th international Conference on Mobile and Ubiquitous Multimedia, MUM 2007, Oulu, December 12-14, vol. 284, pp. 110–117. ACM, New York (2007)
2. Dachselt, R., Frisch, M., Weiland, M.: FacetZoom: a continuous multi-scale widget for navigating hierarchical metadata. In: Proceeding of the Twenty-Sixth Annual SIGCHI Conference on Human Factors in Computing Systems, CHI 2008, Florence, Italy, April 5-10, pp. 1353–1356. ACM, New York (2008)
3. Plumlee, M.D., Ware, C.: Zooming versus multiple window interfaces: Cognitive costs of visual comparisons. ACM Trans. Comput.-Hum. Interact. 13(2), 179–209 (2006)
4. UsiXML (January 15, 2007), `http://www.usixml.org/`
5. Martínez-Ruiz, F.J., Muñoz Arteaga, J., Vanderdonckt, J., González-Calleros, J.M.: A first draft of a Model-driven Method for Designing Graphical User Interfaces of Rich Internet Applications. In: Proc. of 4th Latin American Web Congress LA-Web 2006, Puebla, October 25-27. IEEE Computer Society Press, Los Alamitos (2006)
6. Bederson, B.B., Meyer, J., Good, L.: Jazz: an extensible zoomable user interface graphics toolkit in Java. In: Proceedings of the 13th Annual ACM Symposium on User interface Software and Technology, UIST 2000, San Diego, Cal., United States, November 6-8. ACM, New York (2000)
7. Combs, T.T., Bederson, B.B.: Does zooming improve image browsing? In: Proceedings of the Fourth ACM Conference on Digital Libraries, DL 1999, Berkeley, California, August 11-14, pp. 130–137. ACM, New York (1999)
8. Furnas, G.W., Zhang, X.: MuSE: a multiscale editor. In: Proc. of the 11th Annual ACM Symposium on User interface Software and Technology, UIST 1998, San Francisco, November 1-4, pp. 107–116. ACM, New York (1998)
9. Johnson, B., Shneiderman, B.: Tree-Maps: a space-filling approach to the visualization of hierarchical information structures. In: Nielson, G.M., Rosenblum, L. (eds.) Proceedings of the 2nd Conference on Visualization 1991, IEEE Visualization, San Diego, Cal., October 22 - 25. IEEE Computer Soc. Press, Los Alamitos (1991)
10. Perlin, K., Meyer, J.: Nested user interface components. In: Proceedings of the 12th Annual ACM Symposium on User interface Software and Technology, UIST 1999, Asheville, North Carolina, USA, November 7-10, pp. 11–18. ACM, New York (1999)
11. Appert, C., Fekete, J.: OrthoZoom scroller: 1D multi-scale navigation. In: Proc. CHI 2006, pp. 21–30. ACM Press, New York (2006)
12. Bederson, B.B., Clamage, A., Czerwinski, M.P., Robertson, G.G.: DateLens: A fisheye calendar interface for PDAs. Transactions on Computer-Human Interaction 11(1), 90–119 (2004)
13. Dachselt, R., Weiland, M.: TimeZoom: a flexible detail and context timeline. In: CHI 2006 Extended Abstracts, pp. 682–687. ACM Press, New York (2006)
14. Bederson, B., Boltman, A.: Does Animation Help Users Build Mental Maps of Spatial Information. Submitted to CHI 1999 (1999)
15. Paternò, F.: Towards a UML for Interactive Systems. In: Nigay, L., Little, M.R. (eds.) EHCI 2001. LNCS, vol. 2254, p. 7. Springer, Heidelberg (2001)
16. OMG, `http://www.omg.org` (May 10, 2009)
17. Martinez-Ruiz, F.J., Vanderdonckt, J., Arteaga, J.M.: Context-aware Generation of User Interface Containers for a Mobil Device. In: Mexican International Conferences on Computer Science, IEEE track in Human-Computer Interaction, Mexico, October 2008, pp. 63–72 (2008)

18. http://www.nespresso.com (May 10, 2009)
19. http://www.piccolo2d.org (May 10, 2009)
20. Montero, F., López-Jaquero, V., Lozano, M., González, P.: IdealXML: un entorno para la gestión de experiencia relacionada con el desarrollo hipermedial. In: ADACO: Ing. de la usabilidad en nuevos paradigmas aplicados a entornos web colaborativos y adaptativos, Proy. Cicyt TEN2004-08000-C03-03, Granada (September 2005)
21. Martinez-Ruiz, F.: A Development Method for User Interfaces of Rich Internet Applications, DEA thesis, UCL, Louvain-la-Neuve, August 31 (2007)
22. http://jquery.com (May 10, 2009)
23. Paternò, F., Zini, E.: Applying information visualization techniques to visual representations of task models. In: Proc. of the 3rd Annual Conf. on Task Models and Diagrams, TAMODIA 2004, Prague, November 15-16, vol. 86. ACM, New York (2004)
24. Calvary, G., Coutaz, J., Thevenin, D., Limbourg, Q., Bouillon, L., Vanderdonckt, J.: A Unifying Reference Framework for Multi-Target User Interfaces. Interacting with Comp. 15(3), 289–308 (2003)
25. Dhyani, D., Ng, W.K., Bhowmick, S.S.: A survey of Web metrics. ACM Comput. Surv. 34(4), 469–503 (2002)
26. Miller, G.A.: The magical number seven, plus or minus two: Some limits on our capacity for processing information. Psychological Review 63, 81–97 (1956)
27. Luyten, K., Clerckx, T., Coninx, K., Vanderdonckt, J.M.: Derivation of a Dialog Model from a Task Model by Activity Chain Extraction. In: Jorge, J.A., Nunes, N.J., Cunha, J.F.e. (eds.) Proceedings of the 8th International Workshop on Interactive Systems: Design, Specification, and Verification, Funchal, Madeira Island, Portugal, June 11-13, pp. 203–217 (2003)
28. Chu, H., Song, H., Wong, C., Kurakake, S., Katagiri, M.: Roam, a seamless application framework. J. Syst. Softw. 69(3) (2004)(January 2004)