Tasks Model Composition: Beyond Data, Representing User Activities

Marco Winckler

Interactive Critical System (ICS) Team,
Institute of Research in Informatics (IRIT), Université Paul Sabatier
118 route de Narbonne, 31062 Cedex 9 Toulouse, France
winckler@irit.fr

Abstract. The aim of this paper is to illustrate, by way of a few case studies, the value of model-based task analysis in interaction and design specification of Web mashups. Task analysis is recognized as one fundamental way to focus on the specific user needs and to improve the general understanding of how users may interact with a user interface to accomplish a given goal when using an interactive system. It is argued that some of current challenges to the design and development of Web mashups are commonplace when building task models. So that, in some extension, model-based task analysis can help designers of Web mashups to better understand and describe users' tasks for combining data sources. The cases studies presented hereafter focused on tasks carried out by both developers and end-users of Web mahsups. More than simple illustrations, all cases studies are issued from real life applications.

Keywords: model-based task analysis, Web mashups, task composition.

1 Introduction

The development process for building Web mashups focus on the integration of resources and services available online [5]. So far, most of the research on Web mashups has been devoted to mapping mechanisms allowing data integration, data transformation, and/or representation of processes for composing Web services. However, a deeper understanding of users tasks is required if we want to move forward from simple data integration to more interactive and user friendly Web mashups.

In the field of Human-Computer Interaction (HCI) domain, task analysis is a fundamental way to focus on the specific user needs and to improve the general understanding of how users may interact with a user interface to accomplish a given goal when using an interactive system. In order to provide a representation of user tasks, the HCI community has proposed several task modeling notations [1]. Task models do not imply any specific implementation, so that one can focus on dependencies between activities, availability of resources required to perform tasks and steps users should follow to achieve a task. In the last years, we have been using task model to solve a variety of problems related to the development of interactive systems. Hereafter we summary in three case studies some of the lessons learned that are not alien to the development process of Web mashups.

Q.Z. Sheng and J. Kjeldskov (Eds.): ICWE 2013 Workshops, LNCS 8295, pp. 1–2, 2013.

2 Lessons Learned

Task Refactoring for Helping Developers: Task model are structured around two concepts: task decomposition (a hierarchy) and task flow (order of task execution). When adequately combined, these concepts provide an exhaustive representation of large quantity of information in a single model. However, when applied to real-life systems, tasks notations end up in very large, hard-to-manage models thus making task modeling a time-consuming and sometimes painful activity. In [3] we have investigated structuring mechanisms for task models including encapsulation of task, reuse and task patterns. The lessons we learned is that without the proper structuring mechanism, developers can be easily overwhelmed by their models and lose the focus on the end-users, thus creating ineffective compositions.

Flexible Task Composition to Cope with Many End-Users Needs: During the development of an incident reporting systems [4], we have found out that users could equally achieved their goal by performing different tasks and the choice on which task to perform depended on too many factors that could be decided beforehand, so the system should support all alternatives. Thus alternative was built as a task pattern that is used as a plug & play element in the overall task model. By exploring temporal and logical operators available on tasks models notations, it was possible to accommodate all alternative in a single task model without compromising the required flexibility.

User-Defined Procedures from Task Patterns: In [2] have proposed a Domain Specific Language (DSL) a tool support allowing users to specify their own procedures over the Web by combining task patterns. The underlying task model is orchestrated by the DSL but users don't need to know how to model tasks to use the tool. Moreover, the task composition can be done on the fly accordingly to the user's needs. Two lessons can be learned from this work: i) task models feature patterns useful in many compositions; and ii) compositions can be done on-the-fly by users.

3 Conclusions

In this paper we have provided a few examples of problems that we have solved using model-based task analysis. We suggest that of the solutions presented can provide insights to the development of innovative Web mashups.

References

1. Diaper, D., Stanton, N.A. (eds.): The Handbook of Task Analysis for Human-Computer Interaction, 650 pages. Lawrence Erlbaum Associates (2004)
2. Firmenich, S., Rossi, G., Winckler, M.: A Domain Specific Language for Orchestrating User Tasks whilst Navigation Web sites. In: Daniel, F., Dolog, P., Li, Q. (eds.) ICWE 2013. LNCS, vol. 7977, pp. 224–232. Springer, Heidelberg (2013)
3. Martinie, C., Palanque, P., Winckler, M.: Structuring and Composition Mechanisms to Address Scalability Issues in Task Models. In: Campos, P., Graham, N., Jorge, J., Nunes, N., Palanque, P., Winckler, M. (eds.) INTERACT 2011, Part III. LNCS, vol. 6948, pp. 589–609. Springer, Heidelberg (2011)
4. Winckler, M., Bach, C., Bernhaupt, R.: Identifying User Experience Dimensions for Mobile Incident Reporting in Urban Contexts. IEEE Transactions on Professional Communication 56(2) (June 2013)
5. Yu, J., Benatallah, B., Casati, F., Daniel, F.: Understanding Mashup Development. Internet Computing 12(5), 44–52 (2008)