Designing Interaction Spaces for Rich Internet Applications with UML

Peter Dolog and Jan Stage
Aalborg University, Department of Computer Science,
Fredrik Bajers Vej 7, DK-9220 Aalborg East, Denmark
{dolog,jans}@cs.aau.dk

Abstract. In this paper, we propose a new method for designing rich internet applications. The design process uses results from an object-oriented analysis and employs interaction spaces as the basic abstraction mechanism. State diagrams are employed as refinements of interaction spaces and task models to specify synchronization events and follow up actions on the client and server side. The notation is based on UML.

Keywords: Rich Internet Applications, software design, interaction spaces, state diagrams

1 Introduction

Rich internet applications have been introduced as a response to the limitations in richness that web users experience compared to desktop applications [7, 8]. They employ technologies such as Flesh from Adobe, ActiveX, or recently AJAX technology [11]. The term Rich Internet Applications has been launched first by Macromedia. Later, this technology has also been adopted by others, for example Google and Flickr.

Such applications introduce additional complexity connected with asynchronous communication and synchronization problems as some data are being held and processed at the client side. This influences software engineering methods to build such applications.

In this paper we propose a method for rich internet application design with UML. The method combines interaction spaces and task models [10] with UML based design for adaptive web applications [6]. This combination provides the following advantages:

- Interaction spaces and task models are natural metaphors for designing user interface fragments that a web user will interact with and move beyond traditional web site and web content abstractions.
- The UML statechart diagrams provide means to define user interaction events, synchronization events between client and server as well as synchronization events between fragments of the user interface.

The rest of the paper is structured as follows. Section 2 presents the method for designing rich internet applications with illustrative examples. Section 3 relates the method to the other work in the area. Finally, section 4 provides conclusions and proposals for further work.

2 Design Process and Techniques

Figure 1 is an activity model of our method where:

- **Data Model** describes the problem domain classes needed for application and that are used in use cases and tasks;
- **Use Case Model** provides the context for which the software system is going to be used;
- **Task Model** is a refinement of each use case. It describes the activities that are performed during each use case in terms of a UML statechart diagram.
- **Interaction Space Model** is a refinement of each task model. It describes structural details of corresponding task flows where a user interaction is needed.
- **Guide Model** is a refinement of the task model**.** It provides navigation and synchronization details on user interaction from software behaviour point of view.
- **Mapping to Implementation** maps the design abstractions to the appropriate implementation according to the UML Guide principles [5] employing tagged values, side effect actions and transformations to the running code.

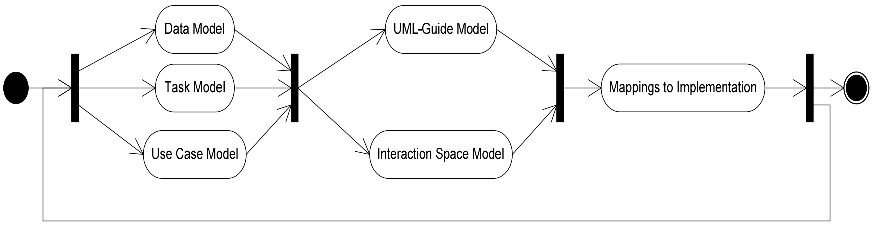

Fig. 1. A design process for rich internet applications

Scenario. The discussion in this section is based on an example of a rich internet application for furniture configuration. The case envisions a website for buying furniture (like IKEA). Consider for example that a user has found a piece of furniture that he considers buying for his living room. He would like to see if it fits into the room with his other furniture and find the model and colour that looks best. He chooses the model page to accomplish this. In that page, he first draws the living room by giving sizes of floor and wall. He also gives the walls and floor colours. Then he puts in furniture sketches of his existing furniture. This is simply done by selecting types and colours from a palette. Now he puts in the new piece of furniture. He determines the various properties of this item, e.g. colour and model, by choosing from the online catalogue. He can change his point of view to see the furniture from different angles. He is able to change the room by moving the existing furniture around. When he is finished, he saves the furniture configuration under his own profile. Once the room is finished, it can be posted to the application server in order to allow the shop to determine whether the requested items are in stock and if necessary recommend alternative options.

Data and Use Case Model. Data and Use Case modelling follows traditional object oriented principles [9] when describing application domain classes, concepts, associations, aggregations and generalization/specialization.

A number of use cases can be identified from the scenario in above such as *Draw a room with walls and floor*, *Make a furniture configuration with existing furniture (Room items)*, *Put in new furniture (Store items)*, *Edit a furniture configuration (change colour etc.)*, and so on. Each use case is described in a textual version and can be depicted in a graphical form on a use case model showing the relationships to the other use cases.

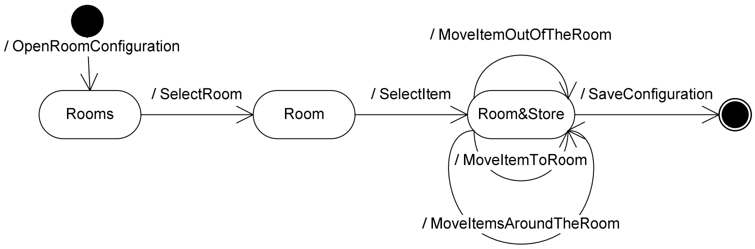

Fig. 2. An excerpt of a task model for furniture configuration

Task and Interaction Space Model. Each use case can also be expressed as a statechart diagram [9]. An excerpt of a task model for our scenario is shown in Figure 3. The transitions are user actions, and the states represent the results of the user actions displayed at a user interface. In accordance with [10], the task models are enriched with *interaction spaces*, forming the elements of user interaction design for each use case. Interaction spaces are conceptual elements which prescribe how particular tasks will be supported by a user interface.

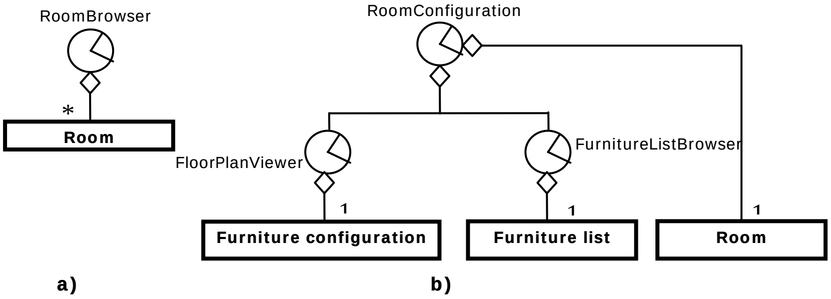

Fig. 3. Interaction spaces for room furniture configuration with data model classes: a) SelectRoom b) SelectItem and move it.

Figure 4 depicts a typical example of interaction spaces with a browser and a view for a problem domain class. Furthermore, each interaction space is further connected to data classes that it is using. For each transition and state in the task model, we consider relevant interaction spaces. For example, we need a *Room browser* that shows all the user's rooms. Selection is made by pointing out one object in the list. We need a *Floor plan viewer* and a *Furniture list browser* to show the contents of the room. In this list, an object can be selected by pointing it out. Moving around should be a drag and drop type of function. These two interaction spaces are parts of an overall interaction space that we call *RoomConfiguration*. Finally, we need an interaction space for saving the furniture configuration.

Guide Model. *Statechart diagrams* are used in the UML-Guide [6] for modelling user navigation in a hypertext; each *state* represents the production of a given information chunk on the device observed by a user, and each *state transition* represents an event caused by user interaction that leads to the production of a new chunk of information. State diagrams therefore provide an abstraction of *hypertext trails*, where each trail can be adapted by taking into account the user background, level of knowledge, preferences and so on [5, 6]. Atomic states, super states, history states, fork and join are additional symbols to describe composition, concurrent execution, remembering, and so on. *Events, guards,* and *side effect actions* are used to specify constraints, triggers, operations for example for synchronizing user and display data as well as adaptation rules.

As each interaction space has already data context through a link to data classes and to the tasks to be performed, it is sufficient to map set of tasks from task model to UML states. Functional dependencies between interaction spaces and their states are modelled as state transitions with parameter passing. These transitions ensure that the user interface will be in consistent and synchronized state. In case of highly interactive activities at the user interface the transitions also ensure that the data is updated according to the user activity.

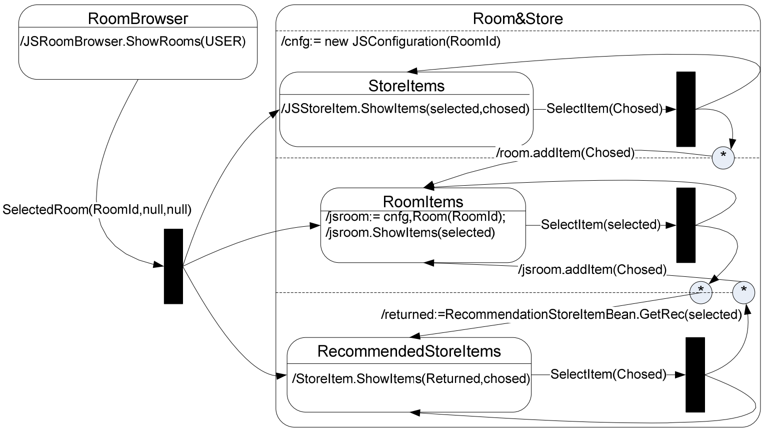
RoomBrowser
/JSRoomBrowser.ShowRooms(USER)
Room&Store
/cnfg:= new JSConfiguration(RoomId)
StoreItems
/JSStoreItem.ShowItems(selected,chosed)
SelectItem(Chosed)
/room.addItem(Chosed)
*
SelectedRoom(RoomId,null,null)
RoomItems
/jsroom:= cnfg,Room(RoomId);
/jsroom.ShowItems(selected)
SelectItem(selected)
/jsroom.addItem(Chosed)
*
*
/returned:=RecommendationStoreItemBean.GetRec(selected)
RecommendedStoreItems
/StoreItem.ShowItems(Returned,chosed)
SelectItem(Chosed)

Fig. 4. An excerpt of the UML guide model for furniture configuration

Figure 4 depicts an excerpt of a navigation model according to UML-Guide. The model is a refinement of the task model presented in the beginning of this section.

There are two dialog steps: *RoomBrowser* and *Room&Store*. The Room&Store incorporates 3 concurrent regions representing three concurrently presented dialog steps: *RoomItems*, *StoreItems* and *RecommendedStoreItems*. Whenever a user selects an item from store items (*SelectItem* event on the transition from StoreItems), the item is placed into the room by synchronization invoked *Room.addItem* function. The model at the client side is reinitialized and client side functions such as *jsRoom.ShowItems()* re-renders the displays considering the added item as part of the scene. More interesting situation happens whenever user selects an item in a room: an event occurs which triggers a synchronizing action with a server of furniture shop and recommended items based on similarity functions and user preferences (*GetRec(selected)* function). Additional parameters can be specified such as booking history and so on.

4 Related Work

A WebML extension for rich internet applications has been proposed in [2] with new units for the client side operation of a web application. In our approach, we propose an alternative UML-based design technique focusing user interaction and behavioral characteristics of navigation with client site business logic and asynchronous communication between server and client.

Task models have been already employed as a means to model complex processes for web applications in WSDM [4]. Similarly to our approach, the tasks are used to describe control flow between user activities and their decompositions to web application internal operations. In our work we use the tasks model together with the interaction spaces. Furthermore, we use decomposition techniques to UML Guide for adaptive navigation design to specify synchronization of different web page fragments and request the data needed for them.

Interactive Dialog Model (IDM) [1] relates to our approach through its focus on user interaction. It is based on a map based technique for user dialog specification on the web connected to data and content.

SHDM [3] uses semantic web conceptual model to specify structural features of the user interface widgets. Widgets are also a central part of the interaction space specifications. They are used to specify parts of the interaction space in terms of data and interaction facilities.

In [12], authors make use of sequence diagrams to synchronize different devices. The technique might be applicable for the rich internet applications as well in combination with the techniques proposed in this paper and in our technique.

5 Conclusion

We have proposed a new UML-based design method for rich internet applications. It is based on the task models, interactions spaces, and web interaction and navigation specification in the UML-Guide. It provides direct means to analyze user interaction as well as asynchronous communication between web client and web server. It reflects the need to have a part of the business logic at the client side. It provides a flexibility with respect to the web page design as this decisions are made later in the design stage according to the UML-Guide principles.

In our further work we plan to conduct larger set of studies about features of this technique. We would like to also see how to further support a designer with some tools which would ease the design refinements.

References

1. Bolchini, D., Paolini, P.: Interactive Dialogue Model: A Design Technique for Multichannel Applications. IEEE Transactions Multimedia, Vol. 8, No. 3, June 2006.
2. Bozzon, A., Comai, S., Fraternali, P., Carughi, G., T.: Conceptual Modeling and Code Generation for Rich Internet Applications. ICWE2006: International Conference on Web Engineering. Palo Alto, California USA. ACM Press.
3. De Moura, S., S, Schwabe, D.: Interface Development for Hypermedia Applications in the Semantic Web. La-Web 2004 Proceedings, Oct. 2004, IEEE Press.
4. De Troyer, O., Casteleyn, S.: "Modeling Complex Processes for Web Applications using WSDM", In Proceedings of the Third International Workshop on Web-Oriented Software Technologies (held in conjunction with ICWE2003), IWWOST2003
5. Dolog, P.: Engineering Adaptive Web Applications. Doctoral dissertation. University of Hannover. March 2006.
6. Dolog, P., and Nejdl, W.: Using UML and XMI for Generating Adaptive Navigation Sequences in Web-Based Systems. In Proc. of «UML» 2003 - Sixth International Conference on the Unified Modeling Language, Oct. 2003, San Francisco, USA, Springer Verlag, LNCS 2863
7. Driver, M., Valdes, R., Phifer, G.: Rich Internet Applications Are the Next Evolution of the Web. Technical report, Gartner, May 2005.
8. Duhl, J.: Rich Internet Applications. White Paper, IDC, November 2003.
9. Mathiassen, L., Munk-Madsen, A., Nielsen, P. A. and Stage, J.: *Object-Oriented Analysis & Design* . Aalborg: Marko, 2000.
10. Nielsen, C. M., Overgaard, M., Pedersen, M. B., Stage, J. and Stenild, S.: Exploring Interaction Space as Abstraction Mechanism for Task-Based User Interface Design. *Proceedings of TAMODIA 06.* Berlin: Springer-Verlag.
11. Paulson, L., D.: Building Rich Web Applications with Ajax. Computer, 38(10):14–17, 2005.
12. Vandervelpen, Ch., Vanderhulst, G., Luyten, K., Coninx, K.: Light-Weight Distributed Web Interfaces: Preparing the Web for Heterogeneous Environments. D. Lowe and M. Gaedke (Eds.): ICWE 2005, LNCS 3579, pp. 197–202, 2005, Springer.