Designing Business Processes in E-commerce Applications

Hans Albrecht Schmid[1] and Gustavo Rossi[2]

[1] University of Applied Sciences, Konstanz, Germany.
schmidha@fh-konstanz.de

[2] LIFIA-Universidad Nacional de La Plata, Argentina
gustavo@sol.info.unlp.edu.ar

Abstract. Business processes play an important role in E-commerce Web applications as they form an important part of the B2C domain and dominate the B2B domain. However, E-commerce application modeling and design techniques have eluded the special characteristics of business processes by treating them just as a special case of navigation. As a consequence, the resulting E-commerce applications have design and usability problems as well as erroneous results from business process execution. We propose a solution to E-commerce Web application design where business processes are considered first class citizens. In this paper we first demonstrate why modeling business processes is important. After a brief introduction, we extend the Object-Oriented Hypermedia Design Method (OOHDM) with business processes. We show that our approach to E-commerce Web application design involving both hypermedia navigation and business processes is easy and clear and does not cause the listed problems.

1 Introduction

In the last few years, a new generation of Web applications has emerged; they differ from purely navigational applications because they use the Web to support and execute business processes and workflows [7][9]. For example, E-commerce applications for rental and reservation services are formed mainly by business processes; auctions and Web shops combine them with navigation**.**

However, both the underlying nature of the Web (as a document-centric information base) and the associated design and implementation tools have not evolved to support the new requirements of E-commerce Web applications. Although a business process has characteristics that are quite different from navigation in hypermedia applications, most Web application modeling and design methods like WebML [2] and HDM2000 [1] treat it as a byproduct of a navigation sequence and do not model and design it explicitly. The consequences for the resulting E-commerce applications are design problems as well as usability problems and erroneous results of business process execution.

This paper presents our approach to support Web application design that allows modeling both business processes and navigation as first class entities. The structure of the paper is as follows: Section 2 shows why business processes should be considered as first class citizens. Section 3 briefly introduces the Object-Oriented Hypermedia Design Method (OOHDM) [10] for Web applications. In Section 4 we extend OOHDM by introducing business processes and activities and show the resulting benefits.

Although we use OOHDM as a basis for our discussion, the ideas underlying the concepts of processes and activities can be equally applied to other design methods such as WebML [2].

2 Rationale for Representing Business Processes in Web Applications

Most electronic stores (like www.amazon.com) include some business process that users execute as a sequence of steps. A good example is the checkout process. During checkout, a user goes through a predefined sequence of activities: he logs in, confirms the items that he is buying, enters the shipping address, selects the delivery options, select the method of payment, etc. Only after all these steps the process is completed successfully.

Suppose that during the checkout process the pages the user accesses are treated as navigational pages (such as CD or book pages). In our example that means the user navigates from the login page to the confirm items page, and then to the shipping address page, and so on. This may cause severe problems, as we will show. The reason behind them is that the semantic of navigation is quite simple:

- The state of navigation is only represented with the current node on your screen, enhanced by the browser collecting a set of visited nodes,
- The user decides freely which node to visit next. That means the designer cannot assume which nodes the user will visit after leaving the current node,
- The navigation history, i.e. which pages a user has visited cannot be inferred from the current node.

It is possible to simulate a business process by navigation only if the process has a very simple and restricted one-way control flow trough a sequence of pages. In this case, navigation may be guided with some special icons and buttons, which indicate that the user should continue the process, on the bottom of these Web pages. But even in this simple case, problems my come up.

One source of problems is a designer not disallowing navigation during the execution of a business process, since providing a possibility for navigation all time may be desirable e.g. in a Web shop.

First, a user may leave the pages of the checkout process without returning because he may get disoriented, or he may consider the guidance about the next step just as an option like it is in navigation, not knowing that the process should be completed. Second, when exploring other pages after leaving the process, the user may cause an inconsistent state in the checkout process. As an example, suppose that during the checkout process the user wants to see again one of the items he is buying. The user then navigates to the product page and clicks the button to add the product to the shopping cart. Will the product be added to the order again? What happens if the user navigates back to the checkout process, will it be resumed or started newly?

Another source of problems lies in using the back button of a browser in the context of a business process. This is perfectly legal in the context of navigation, since it does not matter how the user changes the current page he is in.

But what is the behavior, and what should it be, if the back button is pressed during a business process? Does going to the previous page undo the action that was done on the last page, as most users may believe? Does the re-entry of data on the previous page update the data entered before? While some may argue that this is an implementation problem, the real problem is that the semantics of navigation are completely different from the semantics of business processes.

A business process [7][8] has the following characteristics:

- It drives the user through its activities. This means it defines the set of activities to be executed, and the control flow among them (like the sequence: login, confirm items, select the shipping address, etc.),
- It has a state that consists of the current activity, including if it is active or suspended, and the previously performed activities (in a simple one-way sequential control flow, these can be implied from the current activity).

As a consequence, the activities already executed (i.e. the history), and the subsequent ones are implicit in the current state. Since the current node on the screen does not contain sufficient state information, a business process has to keep its complete state internally during its execution.

The design of business process should allow describing the set of activities and the control flow that form a business process, and if it can be suspended and resumed. In particular, it must allow specifying if the process should be terminated or suspended (and later resumed) when a user navigates out of a business process. If the process is to be suspended, its state must be stored so it can be retrieved when the process is later resumed. Since a business process defines its component activities, it will guide the user through them after resumption.

The above discussion shows that there is a mismatch between navigation and hypermedia (on which the Web is based), and business processes (which play a crucial role in newer Web applications). Despite this mismatch, mature Web application design methods like OOHDM and WebML neglect business processes, or treat them as byproducts of other design primitives. We solve this problem by introducing business processes and their semantics as first-class citizens of Web applications.

3 The OOHDM Design Approach

This section introduces the Object-Oriented Hypermedia Design Method OOHDM [10] together with a CD Web shop E-commerce example that will help us throughout the rest of the paper. OOHDM comprises four activities, namely conceptual, navigation, and interface design and implementation. We summarize two of them, the conceptual design and navigational design activities, since the other are of minor importance with regard to the introduction of business processes.

Conceptual Design: Creating the Application Model

The OOHDM conceptual schema models the application domain without considering specific use-cases. It uses UML as the base modeling language.

Figure 1 shows a part of the conceptual schema for a CD electronic store with a customer, shopping cart, CD, order and other domain object classes**.** The business process „checkout“ is defined as a method of the class Shopping Cart. When the customer initiates the checkout process, the checkout method (which is called from the ShoppingCart node, see navigation design, section 3) creates an order object with the CDs in the shopping cart. An order will contain a set of items, the shipping address, delivery and payment options, etc. When an order is created, these data are obtained from the user.

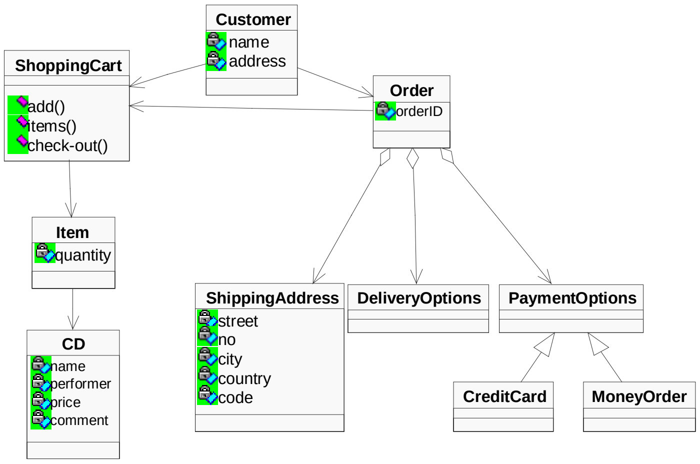

Fig. 1. Class diagram for CD store

Navigation Design: Creating the Navigational Schema

In OOHDM, a Web application is described by a navigational schema as a hypermedia view, which may cover only a subset of the objects and relationships of a conceptual model. The navigational schema contains classes derived from a set of predefined navigational primitives: nodes, links, anchors and access structures, which have the usual semantics of hypermedia applications [10]. Access structures, such as indexes, represent possible ways for starting navigation.

Nodes may contain not only anchors for links, but also behaviors, like „add to shopping cart” in a CD node and „checkout" in the ShoppingCart node (see Figure 2). These methods are triggered from the objects, like buttons, in the graphical interface of a node.

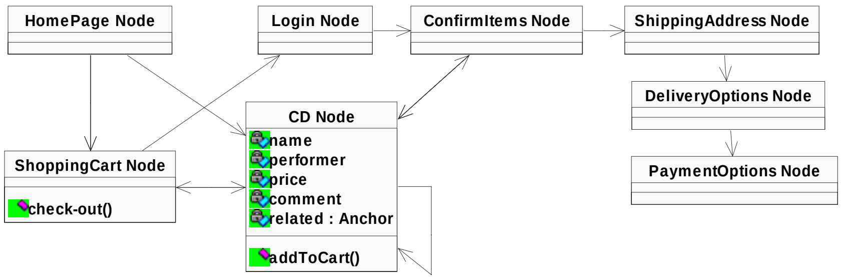

Fig. 2. Navigation Schema for Shopping Use Case

Figure 2 shows on the left the node classes and between them the navigation possibilities as arrows: you may navigate from a customer's HomePage node to a CD node or ShoppingCart node, from the CD node to the ShoppingCart node and vice versa, from a CD to other related CDs by the „related" anchor.
On the top-right, we see the checkout process simulated by a navigation sequence (OOHDM does not use processes as primitives). The Login node is accessed by executing the checkout behavior in the shopping cart (i.e. by pressing the checkout button on its interface). From the Login node, it is only possible to navigate to the ConfirmItems node; from the ConfirmItems Node to the ShippingAddress Node, and so on. Navigation proceeds in sequence, with the only exception is that it is possible to navigate from the Confirm Items node to a CD node and back. The ShoppingCart node and other nodes can be reached from the CD node, with all the consequences described in section 2.

4 Conceptual and Navigational Design of Processes and Activities

Following the OOHDM approach, we introduce in two separate subsections the conceptual definition of business processes and its counterpart in the navigational schema.

4.1 Conceptual Design: Modeling Processes and Activities

To introduce business processes as first class citizens, we partition the conceptual design space in two kinds of objects: entity objects and process objects. Entity objects model permanent entities in the application domain, while process objects model the business processes taking place in that domain. Whether an object should be modeled as an entity or a process is based on different characteristics [7]:

- Entity objects (like customers and orders) have a permanent lifetime and state.

- Process objects do not have a permanent lifetime; their state is temporary. Their instances may be executed in parallel without any problem. They communicate with entity objects sending messages to them.

Figure 3 shows the conceptual schema of the CD store with a design space that is partitioned in entity classes (bottom) and process classes (top). The CheckOut process is composed of several activities like Login Act, ConfirmItems Act, ShippingAddress Act, and DeliveryOptions Act. We consider a business process like Checkout as a parent activity that may consist itself of a set of activities, like Login Act , ConfirmItems Act, etc., following the Composite pattern [4].

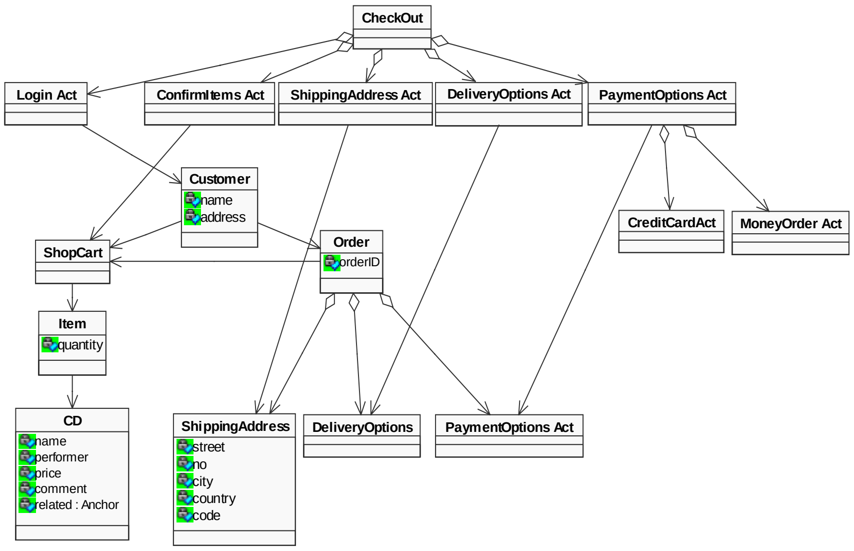

Fig. 3. CD Store with checkout process and activities

A parent activity defines the sequence (the control flow) in which its child activities should be executed, and delegates the work to them. For example, Checkout defines that the activities Login, ConfirmItems, ShippingAddress, Delivery Options, etc. are executed in a strict one-way sequence and invokes them accordingly. Alternatively, CheckOut might define a control flow that gives the user the choice of executing the child activities in any order. The control flow information is not represented in the conceptual schema, since it fits better in the navigational schema.

For a better integration of navigation in business processes, we allow a business process to be suspended and (later) resumed. An activity may finish its processing for several reasons:

- it may terminate if its work is completed,
- it may abort itself since the user cancels the processing,
- it may suspend itself due to a user input like navigation outside of the process.

Each activity has a state, which indicates the current point of control and the current control situation (it may be active or suspended), and other attributes of the activity. A parent activity contains a list of child activities; its state contains information on child activities executed, completed, aborted or suspended. If the control flow among child activities is strictly one-way and sequential, like for checkout, the current child activity is enough as state information. However, checkout with the alternative control flow that gives the user a choice needs additional state information indicating which of the child activities have already been executed.

4.2 Navigational Design: Mapping Processes to Activity Nodes

To model processes and activities, we introduce activity nodes in the navigational schema. We partition the navigational design space in an entity node and an activity node subspace; we map entities from the conceptual schema in navigational nodes (which are nodes as described in navigation design, section 3), and activities in activity nodes.

For example, the navigational schema shown in Figure 4 partitions the navigational design space. On the left, there are the navigational nodes like CD Node, ShoppingCart Node and HomePage Node that were shown in Figure 2. Activity nodes like the LoginAct Node, ShippingAddressAct Node, etc., which map the activities shown in Figure 3, appear on the right. These activity nodes replace the navigational nodes of Figure 2.

Activity Nodes

An activity node represents the output page (HTTP response) and input processing (HTTP request) of an activity; it displays its output and handles its user input. The activity node, or more exactly, its graphical interface contains usually buttons, like *ok*, *commit*, *cancel*, or *next*, which trigger actions related to the input processing of an activity and control the progress of control to a subsequent activity. When an activity allows navigating outside of the process, the associated node contains corresponding anchors for links. From an extended OOHDM point of view, an activity node is a particular node meta-class, which provides an interface for process and activity related behaviors, and another one for navigational links.

An activity node like the LoginAct Node is shown in the navigational schema in the context of the process to which it belongs, which is its parent activity node, in this case CheckOutAct Node. This is indicated by drawing activity nodes within the box of the parent activity node, which may be a process node. Nested process contexts indicate the nesting of composed activities.

Executing Business Processes

When the CheckOut process (see conceptual schema, Figure 3) is started from the navigational ShoppingCart Node by pressing the checkout button, the navigation state is left and the process execution state entered. This is indicated in Figure 4 by an arrow labeled „start" (label not shown in Figure 4) from the ShoppingCart Node to the box of the CheckOutAct Node.

The CheckOut parent activity starts the Login activity (see conceptual schema, Figure 3), which displays its output in the LoginAct Node. After user input, the Login activity receives the user input from the LoginAct Node and processes it. Thus, an activity and its corresponding activity node collaborate closely.

The possible control flow among the activity nodes in a process context is described in a manner that is similar to a UML state diagram (see Figure 4). There is a special initial and final diagram node. Directed edges among activity nodes show the possible flow of control among the activities. In our example, the edges show a one-way sequence from Login over ConfirmItems etc. to PaymentOptions. This contains a nested flow leading from the initial node of the sub-process to its final node, either via CreditCard or via MoneyOrder.

When a business process is terminated, the process execution state is left and the navigation state is entered. In our example CD shop, the application is terminated when checkout is terminated, as Figure 4 shows. But you might as well design the application to enter e.g. the Home Page Node when the check out process is terminated.

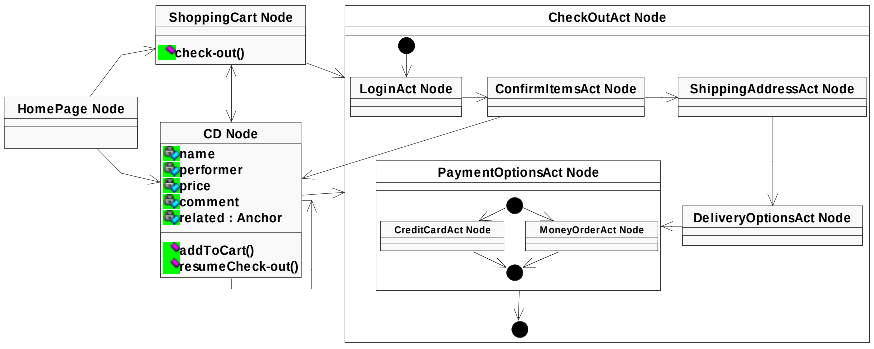

Fig. 4. CD Store navigational schema with activity nodes

Pressing the back button causes the browser to change the displayed activity node, but it does not change the state of the activity. For example, when the back button is pressed during the execution of the ConfirmItems activity, the ConfirmItemsAct Node is left and the LoginAct Node is entered. But the ConfirmItems activity remains active. When the user makes an input to the LoginAct Node, the ConfirmItems activity determines that this is an input not created from its ConfirmItemsAct Node. Thus, it can react in the desired way, one possibility being to tell the user that moving back with the back button was not accepted in a process.

Combining Business Processes with Navigation

You may also suspend a business process to do some navigation, and resume its execution afterwards. The possibility to navigate outside of the process following a link is indicated by an arrow, which goes out from an activity node and leaves the process context box. For example, the arrow from the ConfirmItemsAct Node to the CD Node indicates that you can suspend the CheckOut process and navigate to CDs. When a user follows the link to the CD Node, the ConfirmItemsAct Node suspends the activity, notifies the parent activity CheckOut and returns control to it. CheckOut sets its state to suspended and initiates navigation to the link's target.

The possibility to resume a suspended process is indicated by an arrow labeled „resume" (label not shown in Figure 4) that leads from a navigational node to a business process. For example, the arrow from the CD navigational node to the CheckOutAct Node process context indicates that you can resume the CheckOut process. The arrow does not lead directly to an activity, since the process resumes control as a whole in the state in which it was suspended (i.e. in the corresponding state of the suspended activity). Thus, navigation cannot jump directly to, or within an activity. When the user returns to the CheckOut process, CheckOut knows its suspended state and can resume the suspended activity.

5 Concluding Remarks

Our paper has introduced the modeling and design of business processes in the context of Web applications. Our approach extends OOHDM by introducing process and activity objects in the conceptual design space, and activity nodes and process contexts in the navigation space.

Due to space restrictions, we could not explain navigational contexts defined in OOHDM [10], which may modify the attributes and behavior of a navigational node in a given context. We will show in another paper that navigational contexts may be attached to process contexts to modify the attributes and behavior of a navigational node when accessing it while a process is suspended.

We have shown that considering processes as first class citizens closes the semantic gap between E-commerce business processes and their design notation. In addition, it has important benefits such as:

- Allowing to model and design business processes with a control flow that is more complex than a one-way sequential control flow.
- Solving the usability problems caused by the interplay between navigation and processes, without restricting free navigation.
- Enabling the reuse of activities in different processes. Activities such as customer login or checkout can be reused in the context of different processes, thus improving the quality of design and making implementation more cost-effective.

We have applied the proposed design method to several Web applications in student projects and in cooperation with software houses in real world projects with success. Some of these applications are a customer relation management system for small and medium sized shops and companies, which embodies mainly different business processes, and a cooperative travel agency where users can offer or search for traveling opportunities, which combines different search processes with navigational facilities.

The business processes that are represented in the conceptual schema may be realized and implemented in either an object-oriented or a procedural approach, by putting the state information in the session context of a sevlet.

You may implement a business process directly as a state machine where each state calls for different methods of producing an output page (the response) and reacting on the user input (the request). Alternatively, you may use a state machine framework like Expresso [6]. We have built the WACoF **W**eb **A**pplication **Co**mponent **F**ramework [8] with servlet-based parent and child activity components for a seamless development of a business activity and process implementation from the design presented in this paper.

Acknowledgements. Our thanks are due to Alejandra Garrido and Fernando Das Neves who helped to improve the English, to Anabella Cristaldi who helped preparing the paper, to the International Bureau of the BMBF, Germany, for the Bilateral Cooperation with Argentina support, and to the Ministerium fuer Wissenschaft und Kunst, Baden-Wuerttemberg, for its partial support of this project. This project is also partially supported by SeCTIP, Argentina.

References

1. L. Baresi, F. Garzotto, P. Paolini, and S. Valenti: „HDM2000: The HDM Hypertext Design Model Revisited”, Tech. Report, Politecnico di Milano, Jan. 2000
2. S. Ceri, P. Fraternali, S. Paraboschi: „Web Modeling Language (WebML): a modeling language for designing Web sites”, Proceedings of the 9th. International World Wide Web Conference, Elsevier 2000, pp 137-157
3. J. Conallen: „Building Web Applications with UML”, Addison Wesley 2000.
4. E. Gamma, R. Helm. R. Johnson, J. Vlissides: „Design Patterns. Elements of reusable object-oriented software“ Addison Wesley 1995.
5. I. Jacobson, M. Christerson, P. Jonsson and G. Overgaard: „Object-Oriented Software Engineering”, Prentice-Hall, 1992.
6. P.Pilgrim „Best Practice with Expresso Framework: Using a framework to create a web application”, http://www.theserverside.com/resources/articles/Expresso/article.html
7. H.A.Schmid: „Business Entity Components and Business Process Components”; Journal of Object Oriented Programming, Vol.12, No.6, Oct. 99
8. H. Schmid, G. Rossi and F. Falkenstein „Components for the Reuse of Activities in Web Applications”, in:OOIS 2001, Proc. 7th. Interntl Conf. on Object-Oriented Information Systems, Springer, London, 2001
9. H. Schmid, A. Cristaldi and G.Jacobson „A Business Process Components Framework”, in:OOIS 2001, Proc. 7th. Interntl Conf. on Object-Oriented Information Systems, Springer, London, 2001
10. D. Schwabe, G. Rossi: „An object-oriented approach to web-based application design”, Theory and Practice of Object Systems (TAPOS), Special Issue on the Internet, v. 4 No.4, pp.207-225, October, 1998.