New UML 2.0 based models to design WAP applications

Ricardo Soto De Giorgis
School of Informatic Engineering, Pontifical Catholic University of Valparaíso, Chile
56 32 273762
ricardo.soto@ucv.cl

Nibaldo Rodríguez Agurto
School of Informatic Engineering, Pontifical Catholic University of Valparaíso, Chile
56 32 274095
nibaldo.rodriguez@ucv.cl

ABSTRACT

Wireless mobile applications are becoming more and more popular, mobile Internet technologies, such as WAP (Wireless Application Protocol), are important for anytime, anywhere computing. Although much progress has been made in terms of technological innovation, the modeling activities of WAP applications are still underdeveloped, today practically do not exist models specifically designed for the WAP applications development process.

In this paper we present two new UML 2.0 [1] based models called Decks Navigational Model and Cards Navigational Model, both for the design steps of the application, which we can use to design WAP systems improving the WAP applications development process.

Categories and Subject Descriptors

D.3.3 [**Software Engineering**]: Design Tools and Techniques – *Flow charts*, *Object-oriented design methods.*

General Terms

Design.

Keywords

Software Engineering, Hypermedia, UML, Mobile Internet, WAP.

1. INTRODUCTION

The emergence of mobile Internet technologies and the evolution of wireless devices, that provides huge benefits like anytime and anywhere computing; have increased the development of WAP applications. Though, the WAP applications development process is extremely complex. likewise it exist few models appropriately to this kind of software, driving to the developers to the omission of the structural design of the application. This big difficulty generally gives a low quality application and makes it susceptible of later corrections.

As consequence, the maintenance stage continues being a problem. Not to have the suitable documentation of the application means to transform this process into an exhausting task.

The solution of these problems starts with the creation of a suitable task planning before the application construction. To get this, we need define development methodologies that use models and formal design structures, specially oriented to WAP software.

At the present time WAP system are built using tools that support only the implementation stage, ignoring the important previous process of analysis and design of the structural navigation and interface aspects. Some others approaches have proposed the use of web methodologies, like UWE (UML-based Web Engineering) [2,3,4], OOHDM (Object Oriented Hypermedia Design Method) [5], Conallen [6] or WSDM (Web Site Design Method) [7] to built WAP applications. But there exist outstanding differences between web and WAP systems that carry us to propose models specifically made to design WAP applications.

In this paper we focus in the issues involved in developing appropriated models to design WAP applications. We use the new features provided by UML 2.0 to propose two new UML 2.0 based models called Decks Navigational Model and Cards Navigational Model, which we will use in the design steps of the WAP development process.

This paper is structured as follows: First, Section 2 presents an overview of UML 2.0. Section 3 presents our two new UML 2.0 based models for WAP applications, using a study case. Section 4 a sketch of the study case implementation, and finally presents some concluding remarks and an overview of future work.

2. UML 2.0 METHODOLOGY OVERVIEW

UML is a general purpose notational language for specifying and visualizing complex software, especially large object-oriented projects. UML has emerged as the software language for analysts, designers, and programmers alike, because gives everyone from business analyst to designer to programmer a common vocabulary to talk about software design.

This common vocabulary is provided by means thirteen types of diagrams, divided into two major categories: Structure diagrams and Behavior Diagrams [12].

2.1 Structure diagrams

Class Diagram: a diagram that shows a collection of declarative (static) model elements, such as classes, types, and their contents and relationships.

Component Diagram: a diagram that shows the organizations and dependencies among components.

Object Diagram: a diagram that encompasses objects and their relationships at a point in time. An object diagram may be considered a special case of a class diagram or a communication diagram.

Deployment Diagram: a diagram that depicts the execution architecture of systems. It represents system artifacts as nodes, which are connected through communication paths to create network systems of arbitrary complexity. Nodes are typically defined in a nested manner, and represent either hardware devices or software execution environments.

Package Diagram: a diagram that depicts how model elements are organized into packages and the dependencies among them, including package imports and package extensions.

Composite Structure Diagram: a diagram that depicts the internal structure of a classifier, including the interaction points of the classifier to other parts of the system. It shows the configuration of parts that jointly perform the behavior of the containing classifier. The architecture diagram specifies a set of instances playing parts (roles), as well as their required relationships given in a particular context.

2.2 Behaviors diagrams

Activity Diagram: a diagram that depicts behavior using a control and data-flow model.

Use Case Diagram: a diagram that shows the relationships among actors and the subject (system), and use cases.

State Machine Diagram: a diagram that depicts discrete behavior modeled through finite state-transition systems. In particular, it specifies the sequences of states that an object or an interaction goes through during its life in response to events, together with its responses and actions.

Sequence Diagram: a diagram that depicts an interaction by focusing on the sequence of messages that are exchanged, along with their corresponding event occurrences on the lifelines.

Interaction Overview Diagram: a diagram that depicts interactions through a variant of activity diagrams in a way that promotes overview of the control flow. It focuses on the overview of the flow of control where each node can be an interaction diagram.

Collaboration Diagram: a diagram that gives a specification of how an operation or classifier, such as a use case, is realized by a set of classifiers and associations playing specific roles used in a specific way.

Timing Diagram: An interaction diagram that shows the change in state or condition of a lifeline (representing a Classifier Instance or Classifier Role) over linear time. The most common usage is to show the change in state of an object over time in response to accepted events or stimuli.

2.3 Why UML 2.0?

Now due to UML 1.x was essentially designed for analysis and modeling of small-scale software, in June 2003 OMG standardizes the UML 2.0, completing the definition of this major upgrade of the industry standard modeling notation. UML 2.0 has been revised to better meet the real challenges of systems engineers and software developers by providing better scalability and enhanced support component based development, architecture modeling, and dynamic behavior descriptions.

3. STUDY CASE

In order to explain the use of our proposed models, a generic M-Commerce application will be used. This application will be designed using the standard UML 2.0 extension mechanisms and it will be constructed for mobile systems using WML and PHP.

In this study case we will focus in the design steps of the application. To develop a correctly design we will decompose this stage in three steps:

- Conceptual design
- Navigational design
- Presentational design

Each one has as result a model. Conceptual design produces a class diagram; navigational and presentational design produce our proposed models, the Decks Navigational Model and the Cards Navigational Model, respectively. Figure 1 shows the steps involved and the diagrams/models used in the design stage of the application.

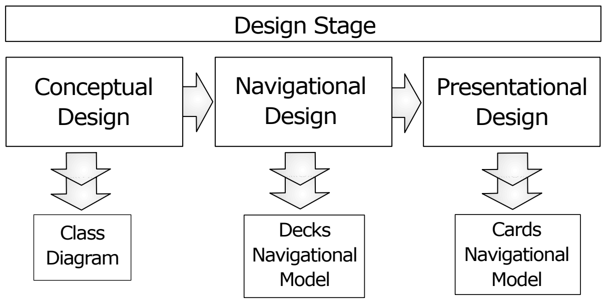

Figure 1. Steps involved and the diagrams/models used in the design stage of the application.

3.1 Conceptual Design

The aim of the conceptual design is to capture the domain semantics, including all the concepts that are relevant for the application and for the different users that have been identified. To achieve this purpose we need to construct a class diagram, which will be the result of the conceptual design step.

3.1.1 Class Diagram

The class diagram gives an overview of a problem domain. To obtain this model, it will be necessary to identify classes, attributes, methods and relations. Therefore, by means of well-known object oriented techniques such associations, compositions, aggregations and generalization a logical structure able to represent in correct way the problem domain it will be defined.

It is important to consider as an essential pre-condition to get an appropriate class diagram, a careful analysis, ideally done using use cases and scenarios [11].

The class diagram shown in the figure 2, models a customer order from a retail catalogue.

The principal class is the Order, associated with it is the Customer making the Payment. A Payment is one of three kinds: Cash, Check, or Credit. The Order contains OrderDetails, each with its associated Item.

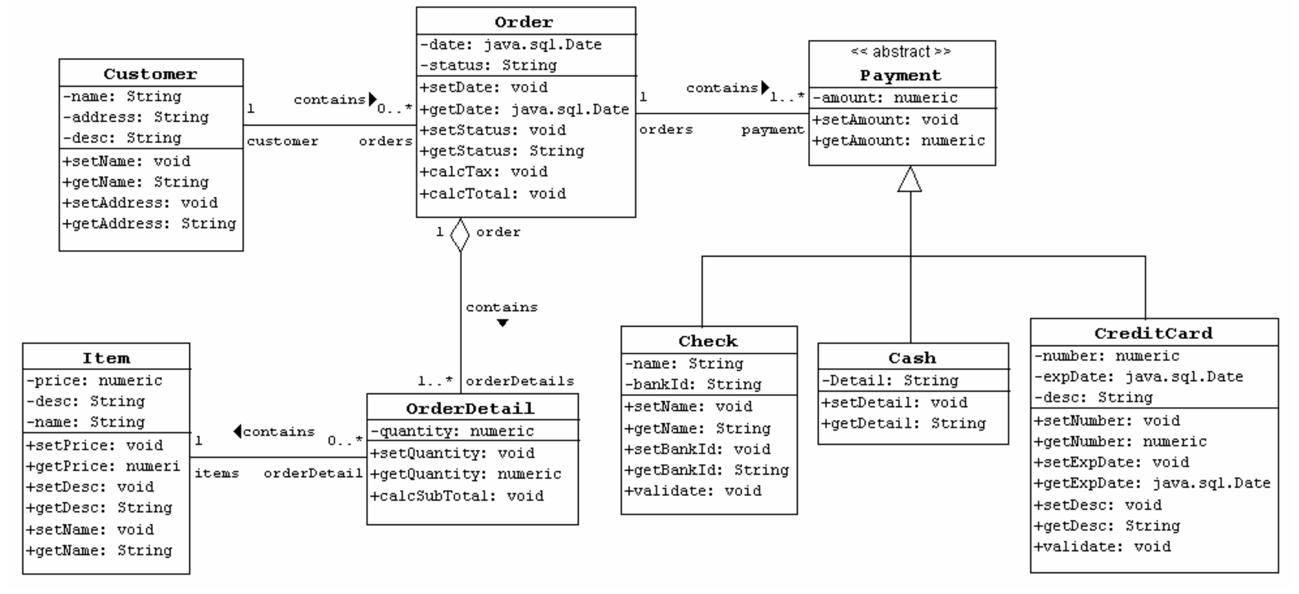

Figure 2. Class diagram of the study case.

3.2 Navigational Design

Navigational design is a critical step in the design of hypermedia applications. One of the main difficulties is to establish a navigational structure that allows the user to navigate in an intuitive way and to avoid him losing the orientation. In order to obtain this, we need design a hierarchical navigation structure, being careful with the nodes and navigation among them. Moreover, we must remember that the navigation model is a valuable document of the maintenance step.

In this step we will propose the use of our new models: The Decks Navigational Model and the Cards Navigational Model. Both models are built with UML 2.0 standard model elements defined according to the standard UML 2.0 extension mechanisms [12].

3.2.1 Cards, Decks and WML.

The markup language used for WAP is WML. WML uses tags and the syntax is stricter and conforms to the XML 1.0 standard [13]. WML pages are called decks. They are constructed as a set of cards, related to each other with links.

The minimum development unit designing WML pages is the card. One deck may contain a lot of cards. Though, in the WAP-browser always it will be displayed only one card. The user can navigate from one card to each other, to visualize card contents.

A set of cards compose a deck. The deck is the minimum transmission unit between the server and the mobile system. When the mobile system receives the deck, it will display the first card of the deck. So, the navigation is done always among the different cards of the deck until a new deck is loaded.

3.2.2 Decks Navigational Model

The Decks Navigational Model is an organized representation of the set of decks of the application. The aim of this model is to achieve an intuitive and hierarchical navigational structure of all decks involved in the application.

Figure 3 shows The Decks Navigational Model of our study case. From this model we can easily understand the navigation between decks.

The starting point of the navigation is the MainMenu Deck, located there the user can selected the item to buy, log-in or register. When the customer has selected the item, he can see its details navigating to ItemDetail. Once the customer has chosen the item he adds it to the ShoppingCart and then he can pay for it navigating to the Payment Deck.

Each node contains the stereotype “<<deck>>” to explicit the kind of the node in the model. Each reference contains the stereotype “<<opens>>”, to explicit that the first deck of the relation calls the target deck allowing the navigation.

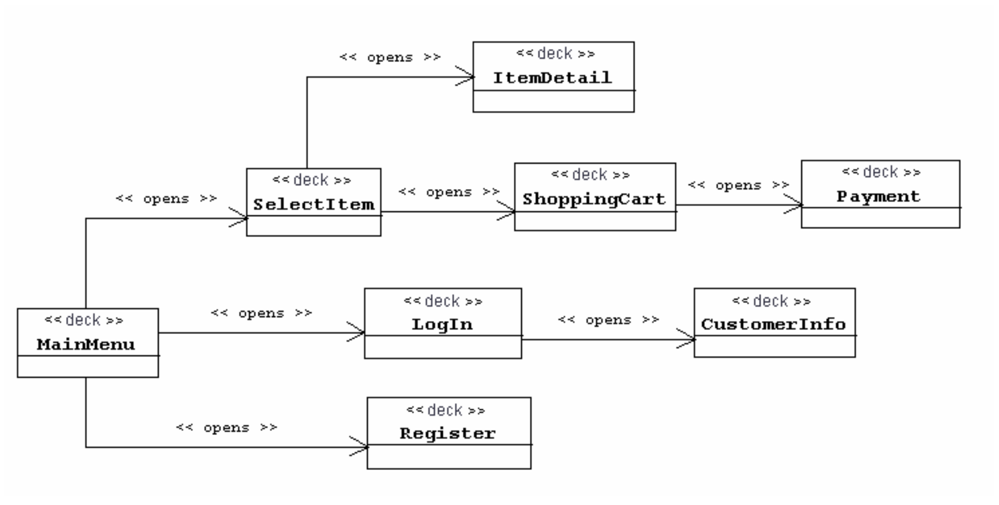

Figure 3. The Decks Navigational Model of the study case.

3.3 Presentational Design

Presentational design gives an abstract representation of the final interface and defines the interaction between the user and navigational nodes. In order to obtain this, we propose the use of The Cards Navigational Model.

3.3.1 Cards Navigational Model

The Cards Navigational Model is an abstract representation of the set of cards of the application. The aim of this model is to achieve a clearly representation of each card to simplify the implementation of the application. This model not consider decisions about details like sizes, colors, fonts or other specific objects of interface, because they belong to the implementation stage.

3.3.2 Stereotyped Objects

Cards contain a lot of elements, each element performs different actions over the system, and all these must be correctly represented in the presentational design. The Cards Navigational Model uses a set of stereotyped interface object to represent clearly all the elements of the final interface of the decks; these stereotypes have been defined according to the standard UML 2.0 extension mechanisms. Table 1 explains the stereotypes for interface objects used in the model.

Table 1. Stereotypes for interface objects

Stereotype	**Name**	**Representation**
<<txt>>	text	Represents a sequence of characters with formatting information
<<im>>	image	Represents an image, generally in wbmp format
<<anc. txt>>	anchored text	Represents an text object with a hyperlink
<<anc. im>>	anchored image	Represents an image object with a hyperlink
<<col. txt>>	text collection	Represents a set of text objects, as a result of a query executed.
<<col. anc txt. >>	anchored text collection	Represents a set of anchored text objects, as a result of a query executed.
<<button>>	button	Represents a clickeable area, which has an action associated.
<<txt. box>>	text box	Represents an input field where the user can introduce information.
<<chk box>>	check box	Represents an input field where the user can check an option
<<col. txt. box>>	text box collection	Represents a set of text box objects
<<col. chk box>>	check box collection	Represents a set of check box objects

Figure 4 shows The Cards Navigational Model of our study case. From this model we can easily understand the objects layout on the deck and the whole navigational structure.

In this model each card and each interface object has an identifier between "{}" characters. Using this identifier it is possible to recognize the target card when starting navigation, for example, in the card MainMenu the interface object "<<anc .txt>>{2} " is identified with 2, which indicates that object interface carry us to the card with identifier 2 (SelectItem). Therefore the complete information of all the system navigation is contained in the model. Furthermore, using the stereotypes, the presentational design of the interfaces also is included in the model.

Presentational design is done by means interface objects, for example, MainMenu card has one interface object text and three interface objects anchored text. The interface object text corresponds to the title of the card, which may be "Retail Catalog". The first anchored text is a hyperlink to the card called SelectedItem, the second anchored text is a hyperlink to the card called LogIn and the last one is a hyperlink to the card called Register.

All cards contains the stereotype "<<card>>" to explicit the kind of the node in the model. Each reference contains the stereotype "<<navigate>>", to explicit that contains a hyperlink to navigate from each card to each other through the reference.

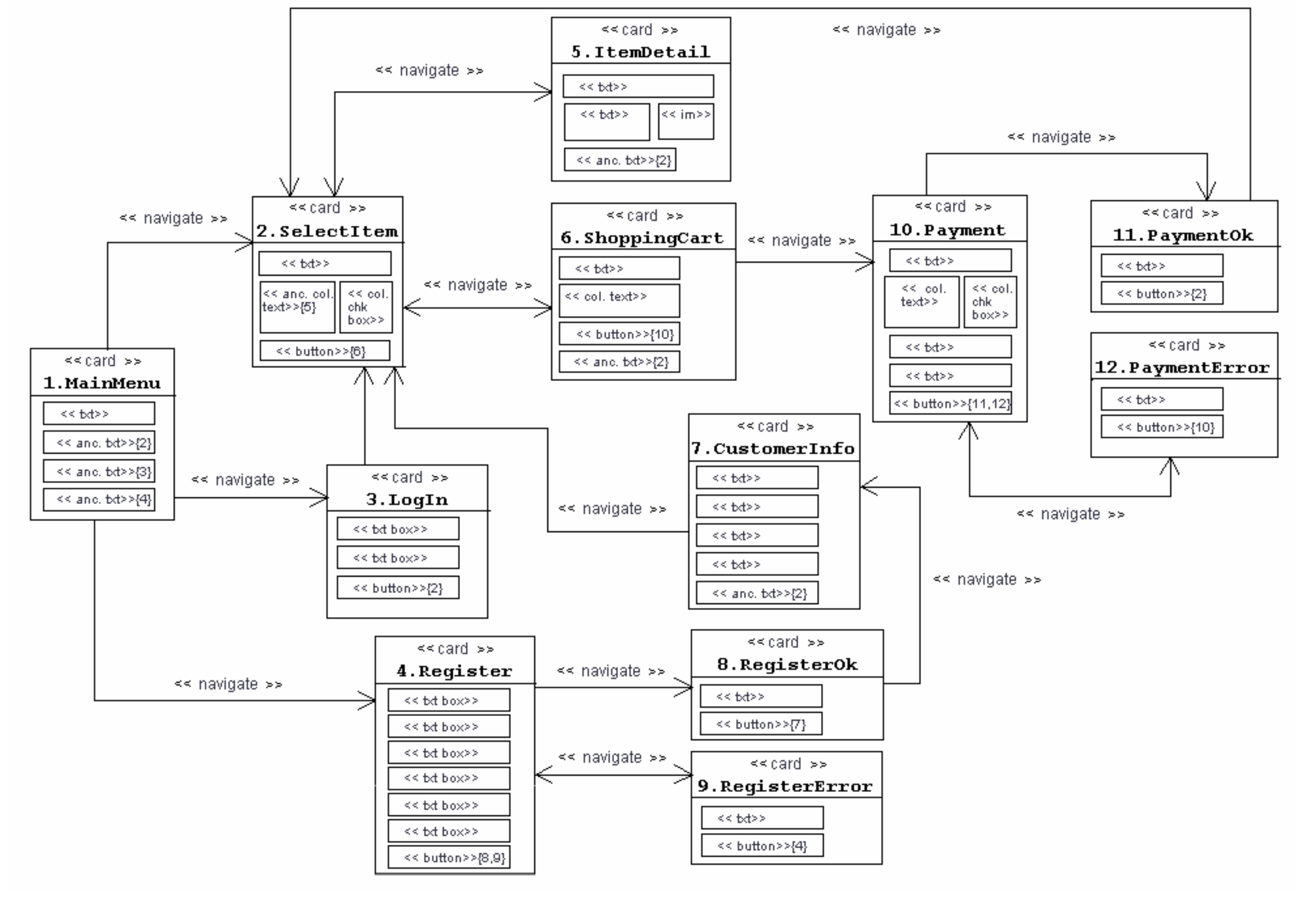

Figure 4. The Cards Navigational Model of the study case.

4. IMPLEMENTATION

In this section a sketch of the implementation of our study case will be given, mainly to understand some advantages that our models provides at the implementing time of the application:

- The clarity of the navigation model allows us to easily understand the system navigation and therefore a fast construction of the links between cards.
- The similarity between the abstract interfaces depicted in the card navigational model and the final interfaces allows us to easily understand the interface objects layout inside the screen
- The simplicity of the model and the abstract interfaces reduces the learning period in being able to use the methodology
- The modeling techniques and notation used in the models presented in the previous section are entirely based on the UML 2.0, a well-known standard and supported by many case tools.
- The previous aspects help us to reduce resources needed and time used in the application development.

Figure 5 shows the similarity between abstract interfaces of the cards navigational model and final interfaces. In addition we can easily compare and understand the navigation depicted by the proposed navigational models

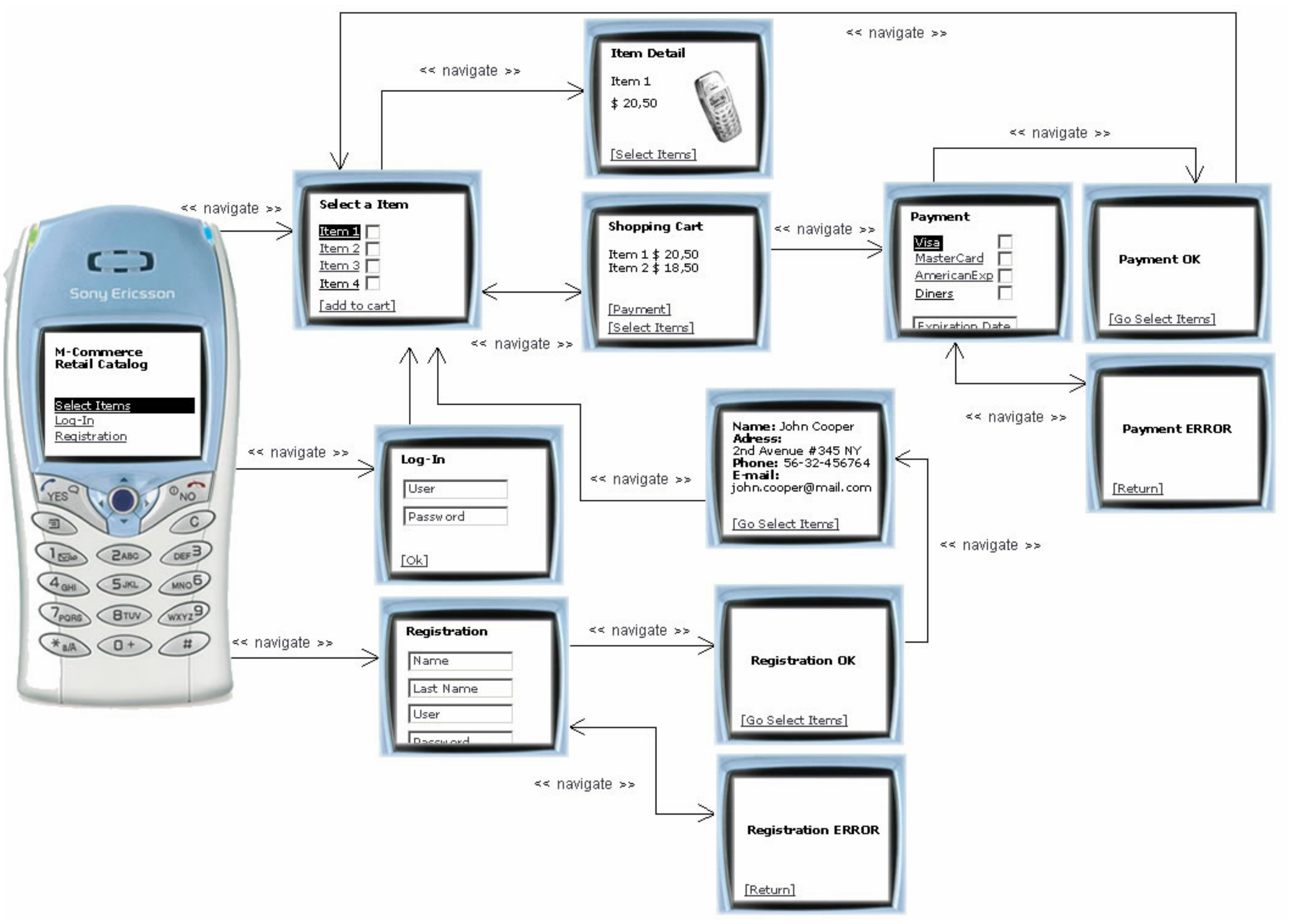

Figure 4. The Cards Navigational Model of the study case.

5. CONCLUSIONS AND FUTURE WORK

WAP applications are inherently complex and require thoughtful design and planning. However hypermedia applications that are systematically designed, using ad-hoc models, require less cycles of improvement and give better results.

In this paper we presented two new UML 2.0 based model called Decks Navigational Model and Cards Navigational Model for the design stage of the WAP applications, which supports navigational and presentational design steps, respectively. The use of these models provides benefits such as clarity, simplicity to understand the system domain reducing resources needed and time used in the application development.

Our future work will be centered on refining the models here presented and working in the other phases of the WAP applications development process: requirements gathering, analysis, implementation, maintenance and quality control. Mainly to obtain experience to develop patterns and a tool CASE that allows us to simplify still more, the whole life cycle of WAP systems.

6. REFERENCES

[1] The Object Management Group (OMG). *"Unified Modeling Language"*.http://ww.uml.org, last updated March 2004.

[2] Rolf Hennicke, Nora Koch. *"A UML-based Methodology for Hypermedia Design"*, Proceedings of the Unified Modeling Language Conference, UML '2000 Evans A. And Kent S. (Eds.). LNCS 1939, Springer Verlag, 410-424.

[3] Nora Koch. "*Software Engineering for Adaptive Hypermedia Applications"*, 2001, PhD. Thesis, Reihe Softwaretechnik 12, Uni-Druck Publishing Company, Munich.

[4] Nora Koch, A. Kraus and R Hennicker. "The Authoring Process of the UML-based Web Engineering Approach", In *First International Workshop on Web-Oriented Software Technology IWWOST'2001*, 2001, Valencia.

[5] Schwabe D., and Rossi G. "An object-oriented approach to Web-based application design. Theory and Practice of Object Systems (TAPOS)", (October 1998), 207-225.

[6] J. Conallen. "Building Web Applications with UML", Addison Wesley, 1999

[7] O. De Troyer and C. Leune. WSDM . *"a User-Centered Design Method for Web Sites"*, Proceedings of the 7th International World Wide Web Conference, 1997.

[8] The Object Management Group (OMG). *"*Model Driven Architecture*"*. http://www.omg.org/mda/, last updated March 2002.

[9] The Object Management Group (OMG). *"*UML 2.0 Infrastructure Specification*"*. http://www.uml.org, last updated March 2004.

[10] The Object Management Group (OMG). *"*MDA Guide 1.0.1 2003*"*. http://www.uml.org/mda/.

[11] D.Leffingwell & D.Widring, "*Managing Software Requirement. A Use Case Approach",* Pearson Education, Boston, 2003.

[12] The Object Management Group (OMG). *"*UML 2.0 Superstructure Final Adopted specification*"*. http://www.uml.org.

[13] WAP Forum DTDs http://www.openmobilealliance.org/tech/DTD/index.htm