A MOBILE USER INTERFACE FOR AN ERP SYSTEM

Karl Kurbel, European University Viadrina, kurbel@uni-ffo.de
Anna Maria Jankowska, European University Viadrina, jankowska@uni-ffo.de
Kamil Nowakowski, European University Viadrina, knowakowski@uni-ffo.de

ABSTRACT

Mobile access to information stored in corporate back-end systems such as Enterprise Resource Planning (ERP), Supply Chain Management (SCM), or Customer Relationship Management (CRM) systems is particularly beneficial for mobile workers. They can interoperate with such a system from anywhere and at any time. However, preparation of content targeted to various mobile devices is a complex task. It requires special content adaptation methods or at least specific design principles. This paper presents an approach for designing mobile user interfaces for information systems equipped with a Web Services Façade. An implementation of mobile access to an open-source ERP system, Compiere, is described with the help of a sample scenario. Access to thin mobile clients is provided through Java 2 Enterprise Edition (J2EE) technology. Open research issues are pointed out in the conclusions.

Keywords: Mobile Front-Ends, Enterprise Resource Planning, Web Services Façade, Compiere

INTRODUCTION AND PURPOSE OF THE WORK

The mobile workforce is rapidly growing and is expected to reach 850 million worldwide in 2009 [8]. This, together with competitive pressure and advances in mobile networks and technologies, is one of the driving forces behind the growing popularity of mobile business-to-business (B2B), employee-to-business (E2B), and business-to-employee (B2E) solutions. Mobile access to back-end systems can substantially reduce operational costs, increase flexibility, and shorten response times.

Although the vision of mobile access to business-relevant information from anywhere and at any time is very tempting, this vision is still hindered by various factors. Major barriers to the adoption of mobile technologies are small screens, low processing power, challenging navigation mechanisms, and a variety of languages and adaptation approaches supported by mobile browsers and Web solution providers (e.g. Mobile Google proxy [12] and the Opera browser with its Small Rendering Technology [15]).

This paper gives an overview of design principles for mobile user interfaces and introduces an architectural approach allowing mobile devices to communicate with an ERP system that is equipped with a Web Services Façade. The paper is organized as follows: In the next section, network standards and categories of mobile devices are discussed. The subsequent section focuses on good design practices and adaptation approaches for mobile user interfaces. Afterwards a brief overview of the J2EE-based architecture of an open-source ERP system providing mobile access is given. The system used in our work is Compiere, an ERP system [5]. The next section describes the implementation of the mobile user interface. Issues for further research and development are pointed out in the final section.

THE HETEROGENITY OF MOBILE DEVICES AND FRONT-ENDS

Designing and developing the user interface is a complex task because of the characteristics of mobile devices and the mobile environment. Mobile networks offer a wide range of data transfer rates; input and output capabilities of mobile handsets vary; and a variety of formats is supported by mobile devices.

Data transfer rates, that can range from 9.6 to 2 Mbps, are a very important factor since users are not willing to wait several minutes for content to be downloaded. Mobile devices are equipped with browsers that support various media formats. In Europe, the Wireless Markup Language (WML) [25] is still in use in simple cellular phones. Due to the low acceptance of WAP, WAP 2.0 introduced XHTML Mobile Profile (XHTML MP) for browser-based user interfaces [22].

Many mobile browsers support HTML so that standard Web pages can be displayed on mobile handsets, at least in theory. However, many Web pages are overloaded with content and graphical elements not suitable for small displays. Additional standards for mobile devices include VoiceXML and

SALT for voice-based applications as well as Java 2 Micro Edition (J2ME) technology for fat mobile clients [20, 16, 19]. Moreover, different mobile devices support different graphics formats like WBMP, BMP, PNG, GIF, and JPEG [9].

At present the most powerful devices have an HTML/XHTML-enabled browser and an additional proprietary browser (e.g. Opera) and they support J2ME. Since different devices understand different languages, content has to be rendered in an appropriate format. This can be achieved either by preparing many versions of the same page in different formats or by generating them dynamically – on the server or with the help of a proxy – and then sending them to the client's device.

Further difficulties in the development of mobile interfaces result from varying properties of mobile handsets, particularly with respect to the input and output capabilities. Developers have to take into account the display sizes, the number of keys specific functionality can be assigned to, and input mechanisms that influence the speed of navigation and data entry.

Categories of existing devices to be considered are Web-enabled cellular phones, smartphones, Personal Digital Assistants (PDAs), handheld PCs, and hybrid devices [11]. Typical cellular phones have a screen size of about 150x150 pixels capable of displaying between 4 and 12 lines of text, and a numeric keypad with 12 buttons and some function keys. Smartphones have larger screens, higher processor power, more memory, and permanent storage. They support J2ME and provide XHTML/WML browsers.

Personal Digital Assistants can be Palm OS-based devices (Palmtops) and Pocket PCs. Pocket PCs come with a regular PC-like operating system installed and are technically more advanced than Palmtops. Screen sizes of PDAs range from 240x320 to 480x640 pixels. Handheld PCs are near to regular laptops, with a display of 480x320 or 640x240 pixels, which can be raised over the full keyboard.

Taking this variety of device features into account, developing mobile front-ends for general usage is a challenging task. Some problems and solutions are discussed in [11]. In the work underlying this paper, we used a UMTS-enabled mobile phone (Nokia 6680) with J2ME and XHTML. It has a high-color display with a resolution of 176x208 pixels and a 12-digit numerical keypad [13].

METHODS FOR DESIGNING AND ADAPTING MOBILE USER INTERFACES

Adaptation can be classified into server-side, client-side, and proxy-based approaches [3]. In client-side adaptation, the formatting of content to the device properties is done on the device itself, usually with the help of Cascading Style Sheets (CSS) [24]. In server-based methods, an appropriate version of content is prepared on a server and subsequently sent to the device. Proxy-based approaches use some intermediary in charge of the adaptation process.

Opera, for example, provides two solutions – a Mobile Accelerator proxy and a special browser for handsets [14]. The browser is a client-side solution using a set of style sheets to squeeze the content to the screen size and scaling images appropriately. The Mobile Accelerator reduces the page size, compresses images and eliminates unnecessary content (e.g. banners). Both solutions yield pages tailored to a mobile screen but pagination or transformation of the site structure is not possible.

Server-side approaches are usually based on XML/XSLT technology [10]. Content is provided in the form of XML documents which are transformed to different markup languages with the help of XSL style sheets on the server and then sent to the end-device. Methods for creating user-friendly mobile interfaces can focus on adapting either the content itself, the style and the layout, or the structure of pages [4].

Content adaptation means in the first place to eliminate unnecessary elements such as images or banners. Alternatively, these elements may be transformed or replaced by a short textual description. In this way the amount of data to be transferred is reduced and some of the vertical and horizontal scrolling can be avoided.

Adaptation of style and layout includes changes in the formatting and presentation of elements; for example, choosing an appropriate font size or color. Avoiding unnecessary breaks or margins and adjusting the spacing are also good ways to improve the readability of mobile pages.

Structural adaptations refer to the flow of pages and the navigation. The flow structure can be adapted by splitting available information into smaller fragments that are displayed on separate pages (pagination). Alternatively, the data may be serialized in some order depending on selected criteria (e.g., important data first).

Proper navigation and control structures can make mobile solutions more comfortable to use. Likewise the user should not have to enter a lot of data. This can be achieved by letting the user select from predefined lists of items with the help of check boxes, radio buttons, or drop-down lists, for example. Intuitive menus can significantly facilitate mobile browsing and decrease the time spent on navigation. Menus can also help to divide the content into separate parts, for example, by arranging information into a tree structure with appropriate captions. Further guidelines for the creation of user interfaces on mobile devices are available. For example, the W3C published a set of rules for preparing HTML for mobile devices [21].

The design of mobile user interfaces depends on the users' requirements. Different studies showed that users want simple navigation structures similar to the menus on Web sites they are familiar with. The amount of text presented on the mobile screen should be small to avoid vertical scrolling and hitting many keystrokes [1, 7].

ARCHITECTURE FOR A MOBILE ERP SYSTEM

Our architecture for the Compiere ERP system with mobile access consists of five layers. It is server based (cf. Figure 1). The services layer is represented by an Oracle database where all relevant information is stored. The application or business logic layer with the business rules is distributed: Some parts are found in stored procedures in the database, some within code modules, and some within the client's graphical user interface.

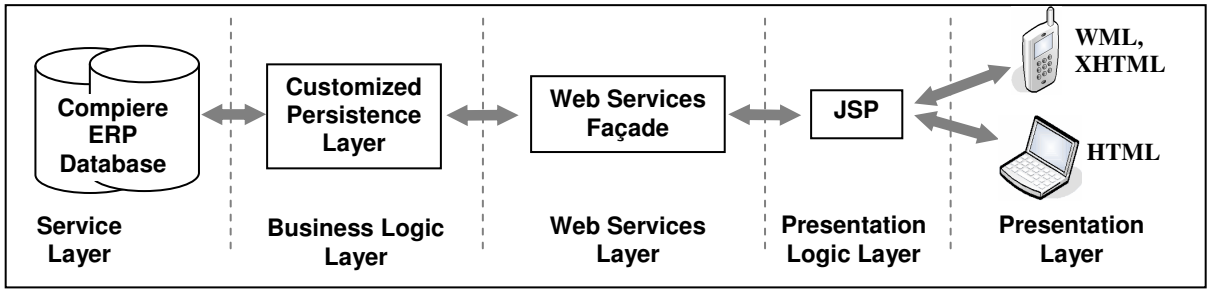

Figure 1. Architecture for ERP System with Mobile User Interface

The Web Services Façade is mainly based on Compiere's customized persistence layer which reflects the underlying database structure. This layer is a mediator between the business logic and the presentation logic. Instead of accessing internal ERP functionality through a proprietary API (Application Programming Interface), the developer can use the Web Services included in the Façade. A Façade is a common design pattern in which an application can be used without knowledge of implementation details behind the Façade [17]. The presentation logic layer is responsible for generating code in an appropriate markup language and adapting content to the device features. It is implemented in the form of JavaServer Pages which use the CC/PP format and DELI library to retrieve device context [6, 23, 2]. The presentation layer consists of browsers capable of displaying HTML, XHTML, or WML. The Web Services Façade consists of different types of Web Services. Some simple services retrieve information about orders, suppliers, business partners, etc. Most Web Services are coarse-grained and not as fine-grained as the original ERP functions.

The runProduction Web Service, for example, initiates a production run based on bills of materials (BOMs). A BOM may contain materials, services, and other BOMs. The Web Service searches for all BOMs involved and computes the required quantities minus the quantities on stock. For each item that exhibits net demand a production order is initialized and a production header – name, description, and date – is created. Then the production plan is generated. All data are saved and the real production process is initiated.

USER INTERFACE FOR ERP SYSTEM

According to different sources, mobile workers account for some 37 percent of the entire workforce [18]. Typical examples of mobile workers are executive and senior managers, sales staff, and service personnel. Each of these groups, despite their

dissimilar functions and tasks, needs instant access to back-end systems. Executives and managers are mainly interested in support for decision making. Sales people visiting customers at their locations may wish to query the ERP system for product information, check available inventory, or place orders. For mobile service personnel it is helpful if they can check maintenance plans or service details.

The main problem addressed in this paper is how to adapt ERP content to the small displays of mobile devices. Taking into account both input and output constraints of such devices, one general rule is to provide relevant information as efficiently as possible. A manager, for example, should only be informed about critical changes or problems (e.g. by SMS or push service) instead of monitoring the ERP system himself by manually entering requests all the time.

In our sample scenario, a sales representative needs information about a customer order. When he is in his office, he can view all necessary details on a Compiere form as in Figure 2. At a customer's place, however, he has only his mobile phone to interact with the system. After logging in, authentication, and authorization, the list of available ERP modules is displayed to him. In the example given in Figure 3, the user wants to see details of the order with ID 80002. The `getOrder` Web Service provides this functionality. The user does not need to type anything but can simply select from drop-down lists with order IDs. The alternative – to type an order ID – is available as a second option. After choosing the ID, the mobile interface communicates with the Web Services Façade through JSP pages. Appropriate methods are invoked and their results are presented on the subsequent pages.

The properties of a mobile device such as screen size, font types and formats, colors, etc. are extracted from device profiles using the DELI library [2]. Depending on the information retrieved, the content is adapted to device properties according to the principles described before and sent back in XHTML/WML or HTML format. In this example, an appropriate font size and font type were chosen (Tahoma 8 in this case).

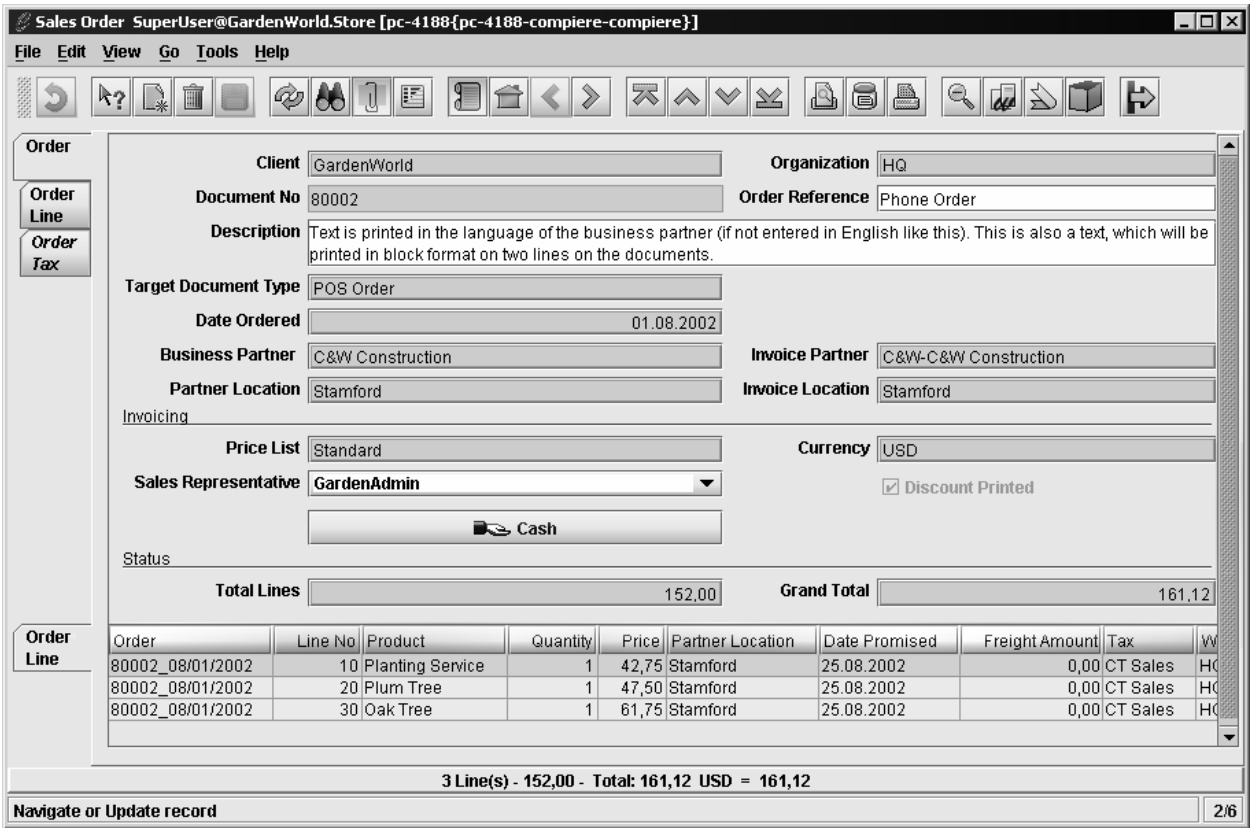

Figure 2. Compiere's GUI Displaying Sales Orders

Figure 3. Nokia 6680 Interface Displaying Sales Orders

Unnecessary line breaks and spaces were eliminated, and the content was grouped with regard to its significance to the user. The most important data are displayed on the first screen. Additional details about order lines, shipment and billing conditions can be obtained on subsequent pages. It should be noted that instead of displaying all data as on a Compiere form, only such data relevant in the specific context are displayed.

The mobile user interface has an intuitive menu structure in the form of links. Pagination and serialization were applied for sorting information before it is displayed. Pagination splits order details into thematically related pieces. Bringing those pieces into a meaningful order while maintaining the importance of information is a means to provide the user with effective navigation mechanisms. Presentation of information follows a well-defined scheme based on a certain set of criteria.

Such a criterion can be, for example, to begin with the most general or abstract concept, and continue with more specific or concrete ones. Pagination and serialization make an interface user-friendlier. An intuitive structure of information and selection mechanisms (e.g. check boxes, radio buttons, or drop-down lists), instead of typing entries enhance the mobile user interface significantly.

OUTLOOK

Although content adaptation methods are discussed in many research papers, a great deal of effort is still needed to automate adaptation. In most present approaches, content from a back-end system that is considered relevant for the user is pre-selected manually. Therefore, it would be very helpful, as a first step, if information specifying the potential importance of items to the user was available in the database. Output presented in GUI forms could then be categorized and described according to its relevance. Weights or significance levels could be assigned to items depending on the category of users.

Compiere has an internal application data dictionary, which is used to describe the structure of data and to identify relationships between the data. Dictionary items define attributes of components such as fields, records, screens, local variables, reports, etc. and additional characteristics such as validation rules and help messages. With the goal in mind to support the reduction of content to the limits of a mobile device, the dictionary could be extended by additional attributes reflecting the importance of GUI items to the user. A Web Service could then query the dictionary and the database, and assemble an appropriate mobile screen from the data retrieved.

The next step is to describe information in the database semantically. Automated adaptation methods might then use this semantic information to identify data that are relevant to a particular category of devices or users. A vision is that future users will be able to access traditional desktop-based back-office applications from their mobile phones in a completely transparent way, with a complex automated adaptation process running behind the curtain.

REFERENCES

1. Buchanan G, et al. (2001). Improving mobile Internet usability. *Proceedings of the 10th International Conference on World Wide Web*, Hong Kong, 673-680.

2. Butler, M. (2002). DELI: A delivery context library for CC/PP and UAProf [online]. Available: http://www.hpl.hp.com/personal/marbut/DeliUserGuideWEB.htm
3. Butler, M., Gianetti, F., Gimson, R. & Wiley, T. (2002). Device independence and the Web. *IEEE Internet Computing, 6*(5)*,* 81-86.
4. Commarford, P. & Cotton, J. (2005). Designing Web content for mobile browsers [online]. Available: http://www-128.ibm.com/developerworks/wireless/library/wi-web/?ca=dgr-lnxw82MobileWeb
5. Compiere ERP System (2005) [online]. Available: http://www.compiere.org
6. Hall, M. (2000). *Core servlets and JavaServer Pages*. Upper Saddle River, NJ: Sun Microsystems Press/Prentice Hall.
7. Hassanein, K., Head, M. (2003). Ubiquitous usability: exploring mobile interfaces within the context of a theoretical model. *Proceedings of the 15th Conference on Advanced Information Systems Engineering (CAiSE '03)*, Klagenfurt/Velden, 180-194.
8. IDC (2005). Worldwide mobile worker population 2005-09 forecast and analysis [online]. Available: http://www.idc.com/getdoc.jsp?containerId =34124
9. Kobayashi, A. (2001). The graphics information sharing platform for mobile computing based on SVG. *Proceedings of the XML Conference & Exposition* [online], Orlando, Florida. Available: http://www.idealliance.org/papers/xml2001/papers/pdf/05-05-05.pdf
10. Kurbel, K., Jankowska, A.M. & Dabkowski, A. (2005). Architecture for multi-channel Enterprise Resource Planning system. *Krogstie, J. et al. (Eds.): Mobile Information Systems II, IFIP International Conference MOBIS*, New York, 245-259.
11. Mallick, M. (2003). *Mobile and wireless design essentials*. Indianapolis: Wiley Publishing.
12. Mobile Google proxy (2006) [online]. Available: http://mobile.google.com
13. Nokia (2005). Nokia 6680 Specification [online]. Available: http://www.europe.nokia.com/nokia/0,8764,70878,00.html
14. Opera (2005). Opera products for mobile [online]. Available: http://www.opera.com/products/mobile/
15. Opera (2005). Small-Screen Rendering [online]. Available: http://www.opera.com/products/mobile/smallscreen/
16. SALTForum (2002). SALT Specification [online]. Available: http://www.saltforum.org/saltforum/downloads/SALT1.0.pdf
17. Shalloway, A. & Trott, J.R. (2001). *Design patterns explained. A new perspective on object oriented design*. Boston: Addison-Wesley Professional.
18. Softlab (2004). The mobile workforce [online]. Available: http://www.softlab.co.uk/fm/142/WP%20The%20mobile%20workforce.pdf
19. Sun (2006). J2ME technology [online]. Available: http://java.sun.com/j2me/
20. VoiceXML (2004). VoiceXML specification [online]. Available: http://www.voicexml.org/spec.html
21. W3C (1999). HTML 4.0 guidelines for mobile access [online]. Available: http://www.w3.org/TR/1999/NOTE-html40-mobile-19990315/
22. W3C (2003). XHTML 2.0. The eXtensible HyperText Markup Language specification [online]. Available: http://www.w3.org/TR/xhtml2/
23. W3C (2004). CC/PP structure and vocabularies 1.0 [online]. Available: http://www.w3.org/TR/CCPP-struct-vocab/
24. W3C (2005). Cascading Style Sheets specification [online]. Available: http://www.w3.org/TR/CSS21/
25. WAP Forum (1999). Wireless Application Protocol, Wireless Markup Language specification [online]. Available: http://www.wapforum.org/what/technical/SPEC-WML-19991104.pdf