Mobile Frontends for Enterprise Resource Planning- An Architechural Approach

Karl Kurbel
European University Viadrina, Business Informatics, D-15230 Frankfurt (Oder), Germany, kurbel@uni-ffo.de

Andrzej Dabkowski
European University Viadrina, Business Informatics, D-15230 Frankfurt (Oder), Germany

Anna Maria Jankowska
European University Viadrina, Business Informatics, D-15230 Frankfurt (Oder), Germany

BACKGROUND OF THE WORK AND TECHNOLOGICAL STATE

An increasingly important requirement for the core information systems that today's organiza-tions rely on is that these systems support the mobile behavior of their users. This trend goes hand in hand with ubiquitous computing [Wei91], i.e. providing access to information and computing power independent of locations and devices.

Technologies for mobile business are maturing, becoming more and more powerful. Some severe restrictions set by the GSM (Global System for Mobile Communica-tion) stan-dard, in particular slow transmission rates (max. 9,600 bits per second), have already been overcome by the HSCSD (High Speed Cir-cuit Switched Data) and GPRS (General Packet Ra-dio Ser-vice) technologies. Those technologies provide theoretical transmission rates of 43.2 kbps and 171.2 kbp/s, respectively. Third generation networks like UMTS (Univer-sal Mo-bile Tele-com-mu-ni-cation System) with up to 2 mbp/s will be available soon and further relax the limita-tions for mobile business imposed by current bandwidths. In Japan, NTT DoCoMo's third generation i-mode service has been launched in 1999 already [Yam02].

In the long run it can be expected that mobile devices will provide similar user interfaces as desktop monitors. With this enhanced frontend functionality, they can serve as mobile clients just like today's laptop, notebook, and personal computers, fostering the convergence of "con-ventional" data processing and telecommunications. This trend raises new challenges for business information systems in general and for enter-prise resource planning (ERP) in particular. A long-term vision for ERP systems is to make all functionality available independent of particular frontend devices - on mobile high-resolution multimedia phones as well as on traditional desktop clients.

Our first step towards this vision is to make *ERP data* available for mobile users. Such data are normally stored in the ERP system's database and managed by a database management system (DBMS). The core questions are thus how to transmit queries by the mobile user from the mobile device to the DBMS used by the enterprise resource planning system; and how to transmit and convert such data from the database tables so that they can be displayed on the screen of a mobile device. The two major aspects of a solution are:

(1) Accessing the content of the database.
(2) Extracting information retrieved from the database and prepar ing it in a device-depend-ent manner.

These two tasks are solved in our approach by a Content Access Engine and a Content Extraction Engine. In the subsequent sections, an architecture around those two engines is presented. The Content Access Engine is in charge of retrieving data from a relational database and representing them in an XML format. The responsibility of the Content Extraction Engine is to detect the type of the user's device and to generate interface-specific forms of the XML data in the respective markup language for the user's display.

This paper is organized as follows. In the next section, the general architecture for mobile ERP, the underlying concepts, and the technologies used are presented. Section 3 illustrates by means of a specific ERP system how this architecture was implemented in a particular case. Some observations and open questions for further research are discussed in the final section.

ARCHITECTURAL DESIGN OF MOBILE ERP

General Considerations for Mobile Applications

Typical application systems today have three major layers: presentation layer, business or application logic layer, and services layer [Bri00, pp. 91-106]. The presentation layer provides the user interface and is responsible for the interaction between the user and the device. The application or business logic layer contains the business rules that drive the given enterprise. The services layer provides general services needed by the other layers, usually including database services, file services, print services, and communication services.

The functionalities of these three layers can be assigned to logical entities called tiers. Mobile and wireless applications are typically deployed with three-tier or multi-tier architectures. Such architectures allow for parallel devel-op-ment of tiers by application specialists and provides flexible resource allocation. They require more planning but reduce development and maintenance costs over the long term by leveraging code re-use and elasticity in product migration [Mye02].

In our work the need to develop an architecture arose from the fact that we had to find an effective way to make ERP system data and functionality available on mobile devices. The major technical requirement for mobile access to an ERP system is presentation of informa-tion in multiple formats. Mobile and wireless devices are equipped with different browsers that support various media formats. It is therefore necessary to deliver the content in different markup languages such as WML, XHTML or HTML [W3C99, WAP02, W3C02]. An architecture should make it easy to add new formats, without changing the existing structure.

Imagine a user with a mobile device browsing a mobile ERP application. He or she navigates through numerous menus and generates requests for the ERP system by choosing from menus displayed on the screen by a mobile browser. In response to such a request the user expects to receive information from the ERP system. This information has to be generated in the specific markup language supported by that browser. Obviously presentation of the data retrieved from the ERP systems is device dependent. It depends on the device's manufacturer, on the display size, on the built-in browser, etc. On the other hand, the data as such are not device dependent, so they can be represented in a metadata format. In our approach, these data - on the way from the ERP system to the user's device - are stored in XML format. XML (eXtensible Markup Language) [W3C00] was chosen because it facilitates data exchange between different systems. XML data can be easily processed, transformed to other formats, reordered, filtered, modified, and extended.

Our architecture for mobile applications is browser based and designed for thin client applica-tions. In this architecture, ERP system functionality can be accessed through mobile and wireless devices. The ERP system as such remains unchanged. The architecture is divided into four tiers. The first tier, the data tier, is represented by the *ERP system's database*. The second tier has the specific application logic of the "mobilization" task encapsulated in the *Content Access Engine*. Application logic is defined as the processes which "do the work" such as requesting data, returning data, formating data, etc., for example building queries from a mobile user's request for information and preparing the results for processing. The Content Access Engine transforms the data retrieved into XML format.

The third tier has the challenging task of device-context aware content delivery to the user, incorporating the presentation logic in the *Content Extraction Engine*. This engine determines the type of the browser and the most important device characteristics, and then tailors the content to significant features of the device. The Content Extraction Engine implements the presentation logic [Sun02b]. The fourth tier consists of different mobile and wireless devices like WAP-enabled cellular phones, PDAs, Palmtops, and Pocket PCs with their respective browsers and GUIs.

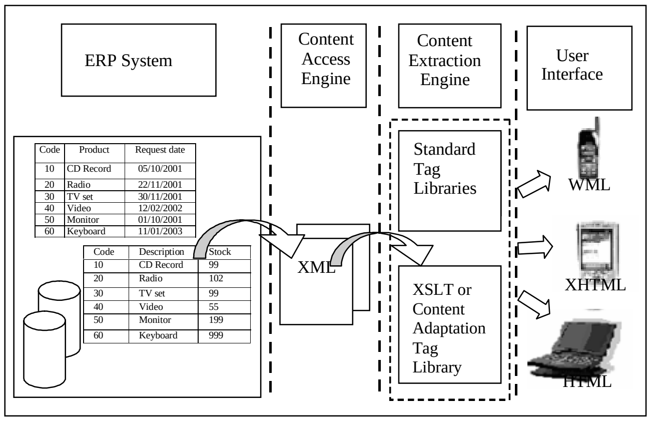
Content Access Engine

The general architecture underlying our discussion is outlined in figure 1. The *Content Access Engine* operates on database tables, as data of an ERP system are usually stored and main-tained by a database manage-ment system (DBMS).

The purpose of the *Content Access Engine* is to transmit data from the ERP database and subsequently convert them into XML-based documents. This procedure consists of several stages. First of all, data gathered from database tables have to be transmitted to the Content Access Engine as shown in figure 2. This task is accomplished by special components implemented as JavaBeans [Sun97]. JavaBeans encapsulate all SQL queries that reflect the physical structure of the requested data in a database [Ger00]. With regard to maintainability and portability of the Content Access Engine, the JavaBeans are the components that have to be modified if the data organization, tables, or dependencies in the database change or if another ERP system wishes to communicate directly with the cache structures mentioned above.

A reasonable and convenient way to store information retrieved from the database for further processing are dynamically built lists (in Java, vectors of vectors). The first row of such a list contains the column

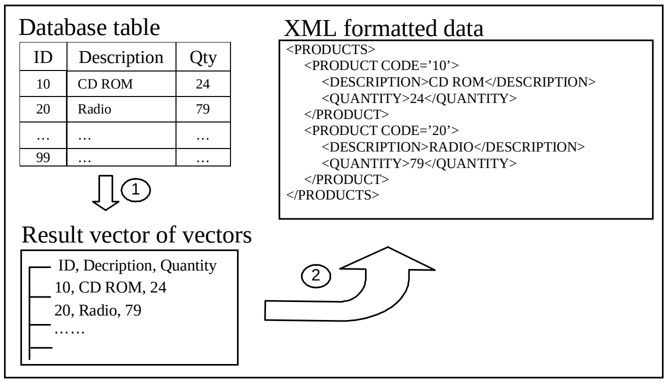
Figure 1: Architecture for mobile ERP



Code	Product	Request date
10	CD Record	05/10/2001
20	Radio	22/11/2001
30	TV set	30/11/2001
40	Video	12/02/2002
50	Monitor	01/10/2001
60	Keyboard	11/01/2003

Code	Description	Stock
10	CD Record	99
20	Radio	102
30	TV set	99
40	Video	55
50	Monitor	199
60	Keyboard	999

Figure 2: Phases of data transfer from ERP system to XML



ID	Description	Qty
10	CD ROM	24
20	Radio	79
…	…	…
99	…	…

```
<PRODUCTS>
   <PRODUCT CODE='10'>
      <DESCRIPTION>CD ROM</DESCRIPTION>
      <QUANTITY>24</QUANTITY>
   </PRODUCT>
   <PRODUCT CODE='20'>
      <DESCRIPTION>RADIO</DESCRIPTION>
      <QUANTITY>79</QUANTITY>
   </PRODUCT>
</PRODUCTS>
```

```
ID, Decription, Quantity
10, CD ROM, 24
20, Radio, 79
……
```

names from the database tables. The column names are used as tag names later, in the next step - the generation of XML documents on-the-fly based on the data in the list (see figure 2). The XML documents serve as input for the Content Extraction Engine.

Content Extraction Engine

A key requirement for mobile applications is to adapt content to the characteristics of a specific device. In our approach this functionality is provided by the Content Extraction Engine. The Content Extraction Engine retrieves information about the devices' features from HTTP headers and performs appropriate metadata transformations.

The most popular way to obtain delivery context is to use the HTTP standard Accept headers [W3C02a]. These headers include the supported media types (MIME types), character sets, content encoding, and languages. Additionally, the User-Agent header contains information about the device manufacturer, the version number, the device hardware and the browser used.

In our approach content adaptation is based on information about device capabilities retrieved from HTTP headers. This information can be obtained with the help of the getHeader method of a request object and is then available during the user's session. The getHeader method is a JavaServlet method. It returns the value of the specified header field as a string [Sun02a, p. 58]. All request header names can be obtained as an Enumeration object from the method getHeaderNames().

Subsequently the Content Extraction Engine determines the form of presentation depending on the relevant features of a device – supported markup language, graphical formats, size of the display area, browser type, and colors displayed.

In the next step, the metadata generated by the Content Access Engine are transformed according to device-specific characteristics. Two components are available for this transformation: tag libraries and transformation objects with XSLT stylesheets [W3C99a].

Tag libraries are reusable modules that can build and access programming language objects and influence the output stream. They usually encapsulate frequent tasks and can be used across applications, increasing the speed and quality of development. Tag libraries have access to all objects available to Java Server Pages, they can communicate with each other and can be nested, allowing for complex interactions within a page.

Custom JSP tags consist of three main parts: the JSP page, the tag handler, and the tag library descriptor. The most important part of a tag – the tag handler – is a Java class that processes the tag. The tag library descriptor is an XML file that groups tags into a library and describes the specifics of each tag, such as its attributes, the tag handler's name, short name, and so on [Sun02, pp. 568-602]. The most popular tags were devel-oped as part of the Apache Jakarta Project and are included in the JavaServer Pages Stan-dard Tag Library (JSTL) [Sun02]. They provide support for tasks such as database access, XML document manipulation, formatting and inter-nationalization.

Although some simple JSP tag libraries for mobile applications have already been provided by vendors, libraries supporting the development of applications for different devices are not yet available. Therefore we developed a special tag library - the *Con-tent Adaptation Tag Library* - for the generation of appropriate markup elements depending on the device context. This library helps to separate the presentation format from the presentation logic. It encapsu-lates most of the functionalities used in HTML, WML, and XHTML pages.

The second component of the Content Extraction Engine are *transformation objects* with XSLT stylesheets. XSLT enables the transformation of XML data into virtually any format. The most popular formats are WML, HTML, XHTML, and SVG (Scalable Vector Graphics) [W3C01]. For the transformations, it is necessary to prepare a number of stylesheets. An XML document then can be parsed and modified according to the respective stylesheet. In our approach the Java Transformation API for XML (TrAX) [Sun02] was chosen to invoke XSL stylesheets from Java programs. TrAX is capable of compiling stylesheets and holding them in memory, thus improving the performance significantly.

AN EXAMPLE OF MOBILIZING A REAL-WORLD ERP SYSTEM

Restrictions and Design Considerations

Significant problems in mobilizing today's ERP systems are caused by low bandwidths of telecommunication networks. It is time and cost ineffective to send larger portions of data to mobile devices. Therefore, in the pre-UMTS era, all processing should be done on an applica-tion server and only the results should be transmitted in a compact form to the mobile device.

Another shortcoming is the low processing power of mobile devices. As a consequence, only simple things like validating input can be done directly on the device while the mobile appli-cation logic must remain on an application server. We took these aspects into account when developing a prototypical solution for a real-world ERP system, *infor:COM* by infor Business Solutions [inf03]. This system aims at small and medium-size enterprises and has a good market penetration in Central Europe.

Before selecting those ERP application domains which appeared worth to be enhanced with mobile access, an empirical study of the state-of-the-art in this field was done [Kur03]. Then the infor:COM system was analyzed with respect to application areas which are both inter-est-ing from a business point of view and feasible taking the technical limita-tions into account.

Content Access and Content Extraction

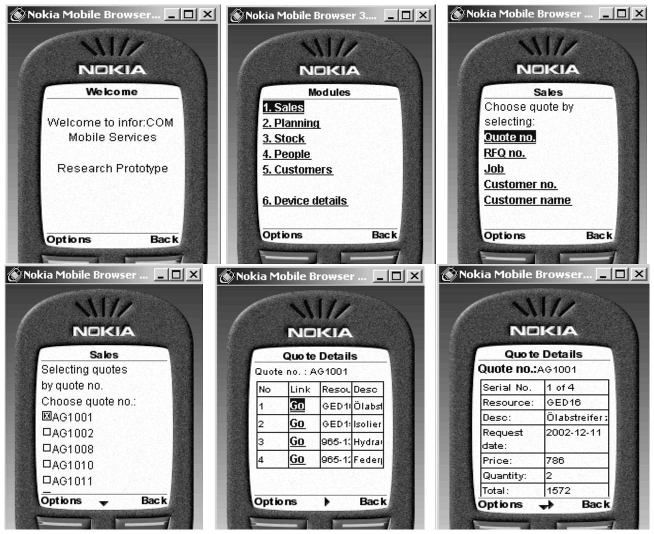
Standard infor:COM applications have a forms oriented user interface, spreading informa-tion all over a conventional screen of a large monitors. Screen content can be quite complex, showing many data and subforms at the same time.

Considering the small display sizes of mobile devices, it is obviously not possible to "trans-late" an existing ERP frontend design to a mobile frontend one-to-one. The developer has to concentrate on important information instead and adopt only the really essential data from the standard screens of the desktop-based application. Usually data have to be distributed across several screens of a mobile device ("cards" in the terminology of mobile browsers).

As an example, processing of quotes in the infor:COM sales module is described. On a stan-dard desktop client, the user generally sees a large form with many menus, submenus, drop-down-boxes, text fields, etc. It is quite obvious that this type of user interface has to be re-designed because it cannot be displayed in the same form on the screen of a mobile device. Therefore the menus in our mobile ERP application are displayed level by level. When the user starts navi-gating on the top level and clicks on a menu item, he or she is redirected to menus detailing the functions of the chosen module. Finally the user reaches a card with the desired functionality.

Figure 3 shows a search for quotes that match some condition specified by the user as a stepwise process implemented for a mobile device. The frontend was implemented with the help of a simulator for mobile devices. Such simulators are contained in toolkits pro-vided by devices manufacturers and other companies [Kur02].

Figure 3: Selecting quotes from ERP database



Welcome

Welcome to infor:COM Mobile Services

Research Prototype

Modules

1. Sales
2. Planning
3. Stock
4. People
5. Customers

6. Device details

Sales

Choose quote by selecting:

Quote no.
RFQ no.
Job
Customer no.
Customer name

Sales

Selecting quotes by quote no.

Choose quote no.:

[✓ AG1001]
AG1002
AG1008
AG1010
AG1011

Quote Details

Quote no. : AG1001

No	Link	Resou	Desc
1	Go	GED1	Ölabst
2	Go	GED1	Isolier
3	Go	965-1	Hydra
4	Go	965-1	Feder

Quote Details

Quote no.:AG1001

Serial No.	1 of 4
Resource:	GED16
Desc:	Ölabstreifer
Request date:	2002-12-11
Price:	786
Quantity:	2
Total:	1572

Assuming that the user looks for a particular quote (quote no. "AG1001"), he or she navigates through a sequence of menus to reach the desired card. The functionality drafted in figure 3 is the same as for the desktop-based infor:COM system. For example, the user can select a certain way of filtering quotes (by quote number, RFQ number, customer no., etc.). From the list of quotes displayed afterwards he or she can select the desired one. If the list of results is long, only a few items are displayed at a time, i.e. the list is stretched over several cards. Likewise the details about quotes are also divided into a number of cards.

While the user notices only the frontend displaying ERP data as requested, the requests and responses are transferred across the overall architecture as described in section 2. In order to look for information the user fills input fields on a mobile screen. In this way he or she deter-mines the criteria for the search. The user's input is treated as parameters and passed to the Content Access Engine where it is further processed behind the curtain.

OPEN ISSUES AND FURTHER WORK

Some starting points for improving the basic components of the architecture described in this paper have become evident in our work. Currently each user request initiates a process of generating XML-formatted data. If the user needs the same data more than once then the entire process of content access and extraction has to be repeated. Or consider two different users requesting the same data. Two separate but basically identical XML documents will be generated by the *Content Access Engine*. A more efficient solution would be to store the XML documents generated last on an application server in a similar way as in a cache. This raises the question whether segments in the cache should belong to a single user only or can be shared by different users. Further research in that direction is necessary.

Another restriction is that the context framework of the Content Extraction Engine comprises only basic information transported in the HTTP headers. One future task will be the integra-tion of comprehen-sive context models. By adding new contextual information to the Content Extraction Engine it will be possible to deliver personalized information, e.g. individual pre-sentation styles according to user preferences.

This enhancement can be achieved by using new standards for contextual information like the CC/PP model developed by W3C [W3C00a] or UAProf introduced by the WAP Forum [WAP01]. Finally, the *Content Adaptation Tag Library* we developed will be enriched in future work with addi-tional functionality. New tags will be developed, e.g. tags for genera-ting forms, input fields, etc. and tags to support other formats (e.g. cHTML, SVG).

In this paper the focus was essentially on accessing the database of an enterprise resource plan-ning system from a mobile device. The next step towards the vision of making ERP functionality available independent of particular frontend devices will be to in-voke genuine ERP functions from a mobile client. While the frontend for this task is quite simple to develop, accessing and/or invoking internal functions of an ERP system is nontrivial unless an API (application programming interface) is available. Since only few ERP systems truly encapsulate their functionalities in an object-oriented manner and provide such interfaces, a lot of reengineering work is necessary.

REFERENCES

[Apa02] Apache Jakarta: XTags library: Tags for working with XML, XPath and XSLT, Version 1.0, 2002. http://jakarta.apache.org/taglibs/doc/xtags-doc/index.html.

[Bri00] Britton, Ch.: IT Architectures and Middleware: Strategies for Building Large, Integrated Systems. Addison-Wesley: Boston, 2000.

[But02] Butler, M.H.: DELI: A delivery context library for CC/PP and UAProf, External Tech-ni-cal Report HPL-2001-260, 2002. http://www-uk.hpl.hp.com/people/marbut/Deli-UserGuide-WEB.htm.

[Ger00] Gertz, M.: Oracle/SQL Tutorial, 2000. http://www.db.cs.ucdavis.edu/teaching/sqltuto-rial.

[Inf03] http://www.infor.de.

[Kur02] Kurbel, K., Dabkowski, A., Zajac, P.: Software Technology for WAP Based M-commerce - A Comparative Study of Toolkits for the Development of Mobile Applications; in: Proceedings of the International Conference WWW/Internet 2002 (IADIS); Lisbon, Portugal; November 2002.

[Kur03] Kurbel, K., Teuteberg, F., Hilker, J.: Mobile Business-Anwendungen im Enterprise Resource Planning: Mobilitätspotentiale entlang der ERP-Funktionskreise; Industrie Management 19 (2003) 1, pp. 72-75.

[Mye02] Myerson J.M.: The Complete Book of Middleware, Auerbach Publ., Philadelphia 2002.

[Nok02] Nokia: WAP June Overview, 2002. http://www.forum.nokia.com.

[Pfe01] Pfeifer, C.: XML Processing with TraX, 2001. http://www.onjava.com/pub/a/onjava/2001/ 07/02/trax.html.

[Sun02] Sun Microsystems: The Java Web Services Tutorial, 2002, http://java.sun.com/web-services/download.html.

[Sun02a] Sun Microsystems: JavaServer Pages™ Specification, Version 2.0, 2002. http://jcp.org/aboutJava/communityprocess/first/jsr152/.

[Sun02b] Sun Microsystems: Designing Enterprise Applications with the J2EE Platform, 2002. http://java.sun.com/blueprints/guidelines/designing_enterprise_applications_2e /DEA2eTOC.html.

[Sun97] Sun Microsystems: JavaBeans, 1997. http://java.sun.com/products/javabeans/docs /spec.html.

[W3C00] W3C: Extensible Markup Language (XML) 1.0 (Second Edition), 2000, http://www.w3.org/ TR/REC-xml.

[W3C00a] Composite Capabilities/Preference Profiles: Terminology and Abbreviations, Working Draft, 2000. http://www.w3.org/TR/2000/WD-CCPP-ta-20000721/.

[W3C01] W3C: Scalable Vector Graphics (SVG) 1.0 Specification, 2001. http://www.w3.org/TR /SVG/.

[W3C02] W3C: XHTML™ 1.0 The Extensible HyperText Markup Language Specification, 2002. http://www.w3.org/TR/xhtml1/.

[W3C02a] W3C: Delivery Context Overview for Device Independence, 2002. http://www.w3c.org/ 2001/di/public/dco.

[W3C98] W3C: Cascading Style Sheets, Level 2 CSS2 Specification, 1998. http://www.w3.org/ TR/REC-CSS2.

[W3C99] W3C: HTML 4.01 Specification, 1999. http://www.w3.org/TR/html4/.

[W3C99a] W3C: XSL Transformations (XSLT) Version 1.0, 1999. http://www.w3.org/TR/xslt.

[W3C99b] W3C: XML Path Language (XPath) Version 1.0, 1999. http://www.w3.org/TR /xpath.

[WAP01] WAP Forum: UAProf, 2001, http://www1.wapforum.org/tech /terms.asp?doc=WAP-248-UAProf-20010530-p.pdf.

[WAP02] WAP Forum: Wireless Application Protocol WAP 2.0, Technical White Paper, 2002. http://www.wapforum.org/what/WAPWhite_Paper1.pdf.

[Wei91] Weiser, M.: The Computer for the 21st Century; Scientific American 265 (1991) 3 (Sept.), pp. 94-104.

[Yam02] Yamakami T.: Leveraging Information Appliances: A Browser Architecture Perspective in the Mobile Multimedia Age; in: Chen, Yung-Chang et al. (Eds.): Proceedings of 3rd IEEE Pacific Rim Conference on Multimedia (PCM 2002), Taiwan, December 2002, pp. 1-8.