SMSFormKit: A SYSTEM FOR PRESENTATION, VALIDATION AND UPDATE OF FORMS FOR MOBILE DEVICES

Carlos Fernández, Juan P. Pece, Daniel I. Iglesia
Facultad de Informática de La Coruña

Keywords: J2ME (Java 2 Micro Edition), XML, XForms, MIDP (Mobile Information Device Profile), CLDC (Connected Limited Device Configuration), WMA (Wireless Messaging API), SMS (Short Message Service), GPRS (General Packet Radio Service), Zlib.

Abstract: Nowadays, many of the services offered by the mobile telephony don't provide to the user a simple and intuitive tool to fill and send information. This way, the users have to code themselves the format of the message, with the corresponding annoyance and the possibility of making errors. One of the solutions to these problems could be provide forms that the user could fill and send from their mobile phones in a transparent way. SMSFormKit is an application which makes easier the creation of new services for mobile phones based on XForms. It has been implemented using J2ME as development environment and provides the following features: forms displaying, validation of the values provided by the user, submission of requests and forms update. Forms are displayed using J2ME high-level graphic elements. This fact guarantees the compatibility with most of the mobile phones in the market. The validation is carried out in the own application (local validation), and this way it is avoided the possible submission of a bad filled form to the server. Thus the server would only receive correct values. Besides, the update and submission of the forms can be done by SMS of GPRS.

1 INTRODUCTION

1.1 State of the wireless technologies

During the last years technologies related with wireless devices have experienced a spectacular impulse. Most of people are already familiarized with the terms mobile phone, PDA, etc. The appearance of these devices is due to the advance of the electronics and, without a doubt, to the spectacular growth of the wireless communications.

As an example of the economic importance of these technologies, we could say that during the last year the mobile manufacturers sold, only in Spain, 17 million of mobile phones (Corredor, 2004), while the number of users at the end of 2004 was 32.2 million (80% of the population in Spain).

Those data point out the great business opportunity which are those services that can be used by that great mass of customers with mobile phone.

More and more those tasks which needed phone operators are being replaced by automatic services that receive messages and carry out the suitable actions without necessity of human participation. As an example we can point out the on-line ticket sales/booking systems, where you can even select the location of your seat. It is a reality in the internet world and now the objective is achieving it in the mobile telephony.

This trend will increase on and on, since the economic benefits that it provides are substantial. Therefore, it seems evident the necessity of systems that automate the interaction tasks with forms for mobile devices.

A detailed explanation of the problem and the proposed solution are respectively exposed in the sections 1.2 and 1.3. Next, section 2 will give a description of our system, including an example of an implemented service. Sections 3 and 4 conclude the paper and describe future research.

1.2 Problem description

The users of these devices demand the possibility of carry out a greater number of tasks and consume new

Fernández C., Pece J. and Iglesia D. (2005).
SMSFormKit: A SYSTEM FOR PRESENTATION, VALIDATION AND UPDATE OF FORMS FOR MOBILE DEVICES.
In *Proceedings of the Second International Conference on e-Business and Telecommunication Networks*, pages 119-124
DOI: 10.5220/0001416401190124
Copyright © SciTePress

services. For both of them it is fundamental the use of forms.

Until the moment, most of the services developed for mobile devices make the user responsible of the correct construction of the format of the SMS message and the data to send. The download of ringtones or mobile games are some examples of those services. You have to type manually a SMS message with a keyword followed by the information to send like this:

RINGTONE song_title

and send it to a mobile number.

As a consequence, the possibility that the user provides incorrect information, the operation fails and the message has to be sent again increases.

Another inconvenience of the actual systems is its inability to validate the values in the fields of the form. Thus, every time an user provides an incorrect value to some of the fields of the form, the server, where the application submits the requests, detects the bad value and asks the user to fix the mistake through some type of communication. Then the traffic of messages increases, as well as the cost to be paid by the customer.

1.3 Developed system

The solution consists on providing to the user services of low cost supported by efficient applications. On one hand, SMS is the most affordable technology in mobile telephony, so its use reduces the costs. On the other hand, the best way to decrease the traffic of messages between the mobile application and the server is the local validation of the values provided by the user. Thus, the server doesn't have to validate them and its computational load decreases.

The solution we propose joins these two features. SMSFORMKIT allows to interpret forms following the XForms standard, show them through the graphic interface of a mobile phone, validate them in the own device, store them in the persistent storage of the phone and carry out petitions and updates of the forms by SMS or GPRS. This way, the cost each time the service is used (in the case you use the service for the first time and also have to download the form) could be only the price of 2 short messages or GPRS short transfers.

At the moment we don't know any other system that implements the same features. The most similar thing we have found is an alphawork from IBM (see (IBM-alphaworks, 2005)).

2 SMSFORMKIT

2.1 System overview

SMSFORMKIT is a system that makes easier the use of new services based on forms for mobile devices. It has been developed using the following technologies: J2ME (Java 2 Micro Edition) (Sun, b), SMS (Short Message Service), GPRS (General Packet Radio Service) and XForms.

The steps of a normal execution of the application are shown in the pictures 1, 2, 3(a), 3(b) and 4:

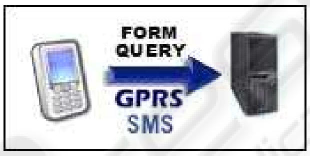


Figure 1: Query for form update

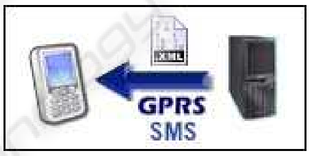
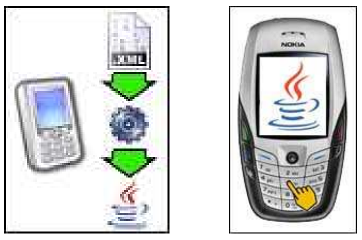


Figure 2: Receipt of the updated form



(a) (b)

Figure 3: a) SMSFormKit parses the form and builds the J2ME graphic interface b) The user fills the fields of the form

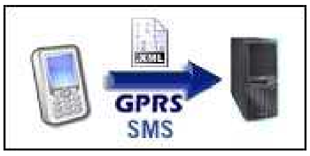


Figure 4: The form is sent to the application in the server

2.2 Graphic interface

The graphic interface of the SMSFORMKIT is very intuitive (allowing an easy interaction with the user) and standard for all the J2ME enabled mobile phones. Those advantages are achieved using high level graphic elements[1] (Muchow, 2003). It was also considered the possibility of using low level elements[2] but the different screen and font sizes of the different terminals were the reasons to discard it.

The class that manages the graphic interface mainly carries out three tasks:

- It keeps a single instance of each one of the screens of the application. It is essential to reduce as much as possible the memory requirements, due to the limitations of the environment. It is implemented a *singleton* pattern, since it is created only an instance of each screen when the application has to show it and later, if the same screen has to be displayed, the same instance is shown. Thus, the number of objects generated by the application decreases.
- It implements the transitions among the different screens of the form and the application.
- It shows the alert messages when the user has carried out an action not allowed, has introduced an incorrect value in a field of the form or the update service is not available.

Every screen of the application except one, uses the high level graphic element *javax.microedition.lcdui.Form* to show its information. This element represents a frame where different *items* can be included. A checkbox list, a radio button list or a text input area are some examples of *item*. Each screen in the application would have an instance of the class *Form* which would only contain one *item*. This way the user will not have to move among the different *items* of the *Form*, making easier the use of the application. The part of the interface that all the services share is designed so that the user should not have to type data, but rather he/she should only select options and press next or back buttons. This fact simplifies the use of the system and makes it more attractive.

SMSFORMKIT begins its execution displaying a screen where the user can select the service he/she wants to use. That screen shows for each service its name and an icon that identifies it (see figure 5).

Once the user has selected a service, it is shown a screen where he/she can choose using the form stored in the own terminal (a previously downloaded form)

[1]As text input areas, checkboxes, radio buttons, etc.

[2]Those elements that the programmer has to draw pixel by pixel

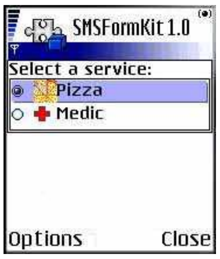


Figure 5: Start screen

or downloading it by SMS or GPRS using the update feature (see figure 6).

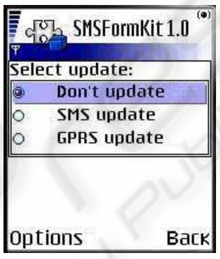


Figure 6: Update selection

After obtaining the XForms document, either from the records of the mobile phone or downloading it again, it is analyzed and the screens corresponding to the fields of the form are built. That set of screens are displayed and the user can navigate among them (see figure 7).

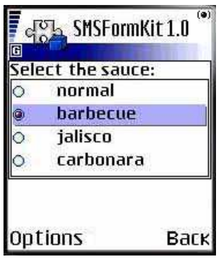
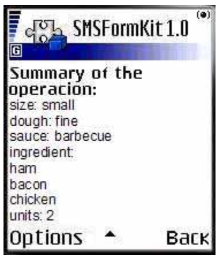


Figure 7: Field to fill in the pizza restaurant service

Finally, when every value for the fields is provided, it is displayed a screen where all those values are shown and the user is requested for a confirmation about their validity (see figure 8). This way, the user can observe if the information is correct or if it is not the desired one and he/she can modify it (in this last case he/she would only have to go back through the screens).

In order to show the fields of a XForms form in the mobile device it has been developed an equivalence among the XForms controls and the J2ME graphic el-


SMSFormKit 1.0

Summary of the operacion:
size: small
dough: fine
sauce: barbecue
ingredient:
ham
bacon
chicken
units: 2

Figure 8: Summary of the operation

ements. This equivalence is shown in the following table:

XFORMS CONTROL	J2ME GRAPHIC ELEMENT
`xforms:input`	*TextField*
`xforms:select`	*ChoiceGroup* + *Choice.MULTIPLE*
`xforms:select1`	*ChoiceGroup* + *Choice.EXCLUSIVE*
`xforms:input` + `readonly=` “`true()`”	*StringItem*
`xforms:secret`	*TextField* + *TextField.PASSWORD*

2.3 Communications

SMSFormKit implements two types of communication:

- Communication by SMS.
- Communication by GPRS.

Every communication in J2ME must be executed in a different thread to avoid the possibility that the application could get blocked if the communication fails.

2.3.1 SMS

SMS was created as a part of the first phase of the GSM European standard and it provides a mechanism to send and receive short messages from a mobile phone up to 160 characters (if a Latin alphabet is used). Some of its advantages are its reliability, its low cost and the guaranteed delivery of messages.

SMSFORMKIT uses the communication by SMS to provide two features:

- **Forms update**: the mobile application sends a SMS message to the application running in the server, requesting the form of a service. Next, it keeps waiting for the server response and finally, when the mobile phone receives the SMS containing the form, the application extracts it, parses it and displays it on the screen of the phone.
- **Queries**: when the user has filled all the fields of the form and they have been locally validated, it is built a message coded in XML (according to the XForms standard) with the provided values and it is sent to the application placed in the server. This application will receive the message, process the data and perform the corresponding action.

Every SMS message will be sent or received from a determined port number (JSR-120, 2003). This number allows to identify the mobile application the messages come from.

Besides, due to its reduced payload, every short message is sent as a binary SMS and compressed as it will be seen in the following section.

2.3.2 SMS compression

The payload a SMS can carry is very reduced. Because of that, it would be useful to maximize that capacity by means of the compression of the data to send. The customer of the services of mobile messaging can simply benefit of the compression algorithms by sending more information per message than with the usual techniques.

SMSFORMKIT uses compression for every SMS it sends. That process consists of two phases:

- Firstly a compression of well-known chains is applied. This compression consists on substituting well-known chains, as the names of the XForms tags, by another shorter chains. For example: `select1` would transform in `s1`.
- Next, the resulting text is applied a binary compression. The first step consists on transforming the chain of text of the message in an array of bytes. The library that provides that compression is an adaptation to J2ME of the Zlib libraries (JCraft, 2003) of data compression without loss, based on Huffman code and the LZ77 algorithm.

2.3.3 GPRS

GPRS, a standard introduced by ETSI, is a system that complements the GSM standard, allowing a better use of the resources. One of the main differences between them is that GPRS makes use of the packet switching technique, while GSM uses the circuit switching one.

SMSFORMKIT implements the communication by GPRS to provide **forms update** and **queries submission** through a HTTP connection (Sun, a):

- The update consists on creating a HTTP connection against a XML file (specific for each service and which will contain the corresponding form), obtaining the number of bytes to read from that file and read the data from it. In this case, it won’t be necessary any decompression type since the XML file is a text file that contains the form without any type of compression. It only has to download the data and then it can already be processed without any manipulation.

- To submit the queries it is created a HTTP connection and the message coded in XML (according to the XFormsstandard) with the provided values is sent to the application placed in the server using POST as request method.

2.4 Forms support

2.4.1 XForms

The forms the application uses follow the standard XForms (Micah Dubinko, 2003). XForms is an XML application that represents the next generation of forms for the Web. By splitting traditional XHTML forms into three parts - XForms model, instance data and user interface - it separates presentation from content, allows reuse, gives strong typing - reducing the number of round-trips to the server, as well as offering device independence and a reduced need for scripting. XForms is not a free-standing document type, but is intended to be integrated into other markup languages, such as XHTML.

XForms has been designed on the basis of several years' experience with HTML forms. HTML Forms have formed the backbone of the e-commerce revolution, and having shown their worth, have also indicated numerous ways they could be improved.

The primary benefits of using XForms are (W3C, 2003):

- XForms improves the user experience: it is achieved by giving immediate feedback to what is being filled in.
- It is XML, and it can submit XML.
- It is device independent.
- Integration with Web services.
- It is easier to create complicated forms.

Our system doesn't implement all the characteristics of the XForms specification, but rather it has been selected a reduced set of properties and elements considered enough (this set can be increased in future works):

- **Elements**: `instance` and `bind`
- **Properties**: `type`, `readonly`, `required` and `constraint`.
- **Controls**: `input`, `secret`, `select` and `select1`.

2.4.2 Parsing

Once the form is obtained, it should be processed to extract the information that contains. The responsible for this task is a small XForms parser that, in turn, uses a XML parser.

The XML parser used is a *pull parser*. XML **pull parsing** (Slominski, 2002) is an alternative to push parsing approach that is very well suited for processing every element of a XML document in a streaming fashion. In push parsers the code of the user is called by the parser when an interesting part of XML input is available and it is a responsibility of the code of the user to keep the state between callbacks. That is the main difference with push parsing: when using a XML pull parser the user code is in control and can pull more data when it is ready to process it.

Due to the limitations of the devices, the mobile application needs a XML parser with small size (hardly some kilobytes) and low memory usage. It has been decided to use pull parsing, and, inside this type of APIs, the kXML 2 one. **kXML 2** (SourceForge.net, 2003) is a compact library which provides a XML pull parser and writer suitable for all Java platforms including the Java 2 Micro Edition (CLDC/MIDP/CDC). Because of its small footprint size, it is especially suited for Applets or Java appications running on mobile devices like Palm Pilots or MIDP enabled cell phones. kXML 2 is licensed under the BSD license. In (Balani, 2003) you can get an example of use.

The result obtained after the parsing of the document is a group of objects, each one of them with J2ME graphic elements and a set of associated validation elements. Those graphic elements will be displayed on the screen, while the restrictions have to be checked for the values provided by the user.

2.4.3 Validation

One of the most innovative aspects in the developed application is the **local validation** in the own device of the values provided by the user for the different fields (making use of the support XForms gives for that purpose).

The term local validation refers to the validation that is carried out using the logic stored in the own device, without having to send the information to an application in a server to be validated there.

Each object of the J2ME application that represents a field will has associated a visual part and a set of constraints. When an user provides a value to a field, if the value doesn't verify the constraints associated to that field, it will be displayed an alert screen with information about the format that the value should have. That alert will be shown again and again until the field would be correctly filled.

This way the user receives an immediate answer about the validity of each given value, avoiding the cost associated to the communications between the client application and the server, in case the validation in the server indicates some error in the data.

2.4.4 Storage

Once we have a downloaded the form, it would be interesting to be able to store it, so we don't have to request it again next time we use the service (for example to order a pizza). SMSFORMKIT provides the possibility of store the forms inside the persistent storage of the mobile phone.

To implement this feature, the system uses the *Record Management System* (RMS[3]). RMS allows to store information among different executions of a J2ME application. This information will be kept in the device in an area of the memory dedicated to this purpose. The amount of memory and the area assigned will depend on each device.

3 CONCLUSIONS

During the development of the system SMSFORMKIT, the following conclusions have been extracted:

- The forms are the simplest and intuitive way so that the users could provide information.
- SMSFORMKIT is a system that provides forms support for J2ME enabled mobile phones.
- It implements the local validation of the values typed by the user. It is a innovative aspect that distinguishes the developed system. The validation is no longer carried out in the server, so the traffic of messages is reduced.
- Besides, it can update the forms of the different services through SMS or GPRS, and store them in the own device.
- The appearance of new services for mobile devices is unstoppable and applications like SMSFORMKIT can increase the number of tasks we will be able to carry out from a mobile phone.

4 FUTURE WORKS

A possible upgrade of the work carried out in the system could consist on developing some of the following features:

[3]The package to use would be *javax.microedition.rms*

- The increase of the number of XForms elements recognized by the application, as well as provide it of XPath support.
- The development of a mechanism of incremental update of the forms.
- The development of new ways of forms updating like the discharge through Bluetooth.
- The development of an option for the discovery of new services: this option would consist on notifying the user the appearance of new services (like a ticket sales system).

ACKNOWLEDGMENTS

This work has been supported by the Xunta de Galicia, grant number PGIDITSIN105002PR.

REFERENCES

J2ME MIDP 1.0 Profile Specification. http://java.sun.com/j2me/.

Java 2 Platform, Micro Edition (J2ME) Home Page. http://java.sun.com/j2me/.

Balani, N. (2003). Using kxml to access xml files on j2me devices. *http://www-106.ibm.com/developerworks/edu/wi-dw-wi-kxml-i.html.*

Corredor, J. (2004). Móviles en españa: hasta aquí hemos llegado. *http://www.baquia.com.*

IBM-alphaworks (2005). Ibm forms for mobile devices, 2004. information available at: http://www.alphaworks.ibm.com/tech/ifmd, last accessed 20 june 2005.

JCraft (2003). *JZlib Home Page.* http://www.jcraft.com/jzlib/.

JSR-120 (2003). *Wireless Messaging API (WMA) for Java 2 Micro Edition Reference Implementation, version 1.1.* Java Community Process.

Micah Dubinko, Leigh L. Klotz, R. M. T. V. R. (2003). Xforms 1.0. *W3C.*

Muchow, J. (2003). J2me 101, part 1: Introduction to midp's high-level user interface. *IBM developerWorks.*

Slominski, A. (2002). *XML Pull Parsing Home Page.* http://www.xmlpull.org/.

SourceForge.net (2003). *kXML Home Page.* http://kxml.sourceforge.net/.

W3C (2003). Xforms 1.0 frequently asked questions. *W3C.*