CAM: A Mobile Paper-based Information Services Architecture for Remote Rural Areas in the Developing World

Tapan S. Parikh
Department of Computer Science and Engineering
University of Washington, Box 352350, Seattle, WA 98195-2350
tapan@cs.washington.edu

Abstract

During our work with microfinance groups in rural India, we found that paper plays a crucial role in many local information practices. However, paper-based record keeping can be inefficient, so we need to link paper with the flexibility of modern information tools. A mobile phone can be the perfect bridging device. Here we present the CAM mobile document processing system, in which a camera-equipped mobile phone is used as an image capture and data entry device. Our system is able to process paper forms that contain CAMShell programs - embedded instructions that are decoded from an electronic image. By combining 1) paper, 2) audio, 3) numeric data entry, 4) narrative scripted execution and 5) asynchronous connectivity, we have synthesized five years of design experience into a usable and locally harmonious system that has the potential to impact billions of people.

1. Introduction

During our work with community microfinance groups in rural India, we found that paper plays a crucial role in many local information practices [1]. It is used ubiquitously as a method of data storage, exchange and establishment of trust between two transacting parties. It is a medium that the local people own and trust, and provides a greater sense of security than rented or borrowed appliances.

The mobile phone has been shown to be the most likely modern digital tool to support economic development in developing nations. The growth of mobile phone use in China, India and Africa resembles the growth of the Internet in the developed world in the 1990s. As shown in the example of Grameen Phone, if a mobile phone is shared by a group of people, even the poorest communities can afford one.

In this paper we introduce mobile document processing as an accessible and locally harmonious way to provide information services to the developing world [2]. In our system a camera-equipped mobile phone is used for image capture and data entry. Document interaction is specified using CamShell - a narrative document-embedded programming language that can allow developers to add interactive audio, data entry, validation, processing, and networking instructions to otherwise inert paper documents.

2. The CAM Architecture

CAM unites a user interface, a programming language and a delivery mechanism for accessing networked information services. Like the web, our goal is to allow many different services to be developed and deployed on a common infrastructure.

2.1. CAMBrowser

The CAM client application is called the CAMBrowser, and has currently been implemented for Series 60 camera-equipped mobile phones. CAMBrowser is designed to process specially designed CAMForm documents. CAMForms contain visual codes - two-dimensional data glyphs containing up to 76-bits of data that can be decoded from a camera image [3]. Visual codes serve as references to interactive content embedded within CAMForms.

CAM interaction consists of two primitives. The user can either *scan* codes from low-resolution images taken by the viewfinder in real time, or *click* codes by taking a high-resolution image using the joystick button. Our focus is on designing a simple and intuitive interaction model, with well-defined affordances between actions.

2.2. CAMShell

CamShell is the programming language that is used to define the interaction with visual codes embedded in the form. There is one visual code in every CAMForm that serves as a *form identifier*. This code always has a 0 as the lowest-order bit. The next 4 bits

Proceedings of the 2005 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05)
0-7695-2443-5/05 $20.00 © 2005 **IEEE**

identify the form to the CAMBrowser application. The form identifier is used to load the *form description schema*, an XML file containing metadata about the form and specifying the underlying data elements (including any default or pre-existing values). The protocol that should be used to load the schema file (currently either HTTP, bluetooth or the local filesystem), and the required information to find the schema (a network address or a file name), is included in the remaining bits of the form identifier.

Visual codes with a lowest-order bit of 1 are *action codes*. Each action code invokes a separate callback function when it is *clicked* or *scanned*. These callbacks are mapped to a set of code entry points specified as XML elements in the form description schema. Each element contains the executable code that should be invoked when the callback is activated. The executable code itself is written using a simple scripted programming language that includes support for function calls, control flow, arithmetic and a few basic datatypes [4].

CAMShell also provides an API for accessing the mobile phone's functionality - including functions to access the phone's user interface, networking and telephony. Each callback function can contain any number of sequential actions, some of which may be only executed conditionally. Conditionals are useful when performing form validation. An example callback function is given below.

```
<function name="u_click" params="param1">
seq = input_int("Please input Form ID");
if (!seq) return false;
uri="http://abc.com/reload.php?"."seq=".seq;
return http_get(uri);
</function>
```

2.3. CamForms

Although there is nothing in the specification of visual codes or CAMShell that dictates how CAMForms should be organized, there is a convention that we have been developing to build a consistent and understandable metaphor for users. CamForms are conceptually organized into the following three sections:

The *header* contains the form identifier code and other codes containing static data (for example identifying the sequence number of a particular form instance). Header codes need to be clicked only once at the beginning of a form interaction.

The *body* contains input codes that are like HTML input tags. These are co-located with form fields, and are used to transcribe data from the document using the phone's keypad. To edit a form field the user must click on the input code, which brings up an editable dialog window. When a set of input codes is clicked together in one image, their respective dialogs are displayed in sequence. When the CamBrowser scans a visual code, the value returned by the associated scan callback function is shown on the screen. This is usually the value associated with the particular field.

Buttons are codes identified by text and/or icons that allow the user to perform various actions. Some example actions include submitting the form data to a web server, reloading the data from an online source or voiding the currently stored data for the form.

3. Applications

We believe that CAM is suitable for a rich set of applications in rural India and elsewhere. We have met banks, private companies, NGOs and governmental agencies that have expressed interest in using the CAM framework. Below are some of the applications that we are currently exploring:

- Sales and inventory tracking
- Health data collection
- Transaction processing
- Mobile cash register
- Accounting for microfinance groups
- Paper / digital postcards

4. Conclusion

In this paper we have described a mobile interaction framework for bringing information services to the developing world. By combining 1) paper, 2) audio, 3) numeric data entry, 4) narrative scripted execution and 5) asynchronous connectivity via the medium of an increasingly ubiquitous mobile device (the mobile phone), we have synthesized the observations of five years of rural design experience into a usable and locally harmonious system that we believe has the potential to impact the lives of billions of people.

.

5. References

[1] Parikh, T., Ghosh, K., Chavan, A., Syal, P. and Arora, S. Design Studies for a Financial Management System for Micro-credit Groups in Rural India. In Proc. CUU 2003, ACM Press (2003), 15-22.

[2] Parikh, T. Using Mobile Phones for Secure, Distributed Document Processing in the Developing World. IEEE Pervasive Computing, vol. 4, no. 2, 2005, pp. 74-81.

[3] Rohs, M., and Gfeller, J. Using Camera-Equipped Mobile Phones for Interacting with Real-World Objects. In Advances in Pervasive Computing, Austrian Computer Society (2004), 265-71.

[4] Simkin language, May 2005. http://www.simkin.co.uk