CAM: A Mobile Interaction Framework for Digitizing Paper Processes in the Developing World

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

ABSTRACT
During our work with community 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 phone is used as an image capture and data entry device. Our system is able to process paper forms containing *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 our experience into a system that we believe is well-suited for an important set of users and applications in the developing world.

Keywords: rural development, paper user interface, document processing, visual codes, literacy

INTRODUCTION
Recently we have seen a growing interest in the topic of the *digital divide* - between people that take advantage of modern digital tools and information services for personal or professional purposes, and those that do not. This has given rise to the optimistic notion that if digitally disenfranchised people can adopt information technologies in a sustainable way, they can achieve many development objectives, particularly in rural areas of the developing world. Termed *Information and Communication Technologies (ICTs) for Development*, this vision carries a broad and pressing mandate - allowing billions of people access to services as important and varied as health care, education, financial and governmental services.

During our work with community microfinance groups in rural India, we found that paper plays a crucial role in many local information practices [15]. 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. Based on these observations, we conclude that paper must be retained as an integral part of this information ecology.

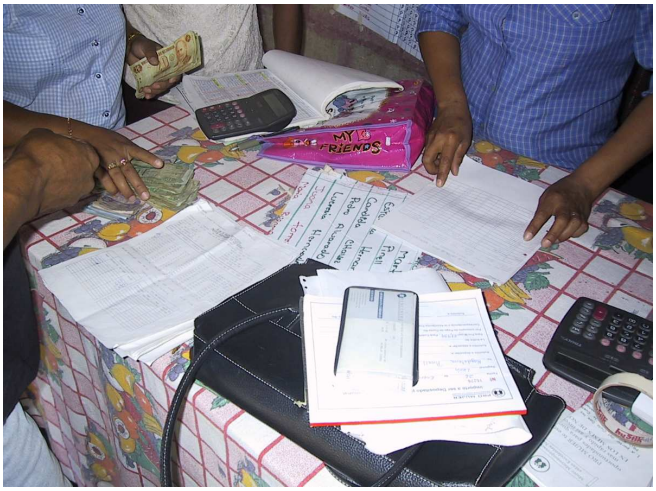

Figure 1: Paper-based information processes are ubiquitous in the developing world.

However, the introduction of some technology is desirable. Not only are purely paper-based processes inefficient, but they do not easily lend themselves to certain kinds of analysis. This can be a serious impediment to information management in developing countries, particularly in the government, financial and health sectors.

The mobile phone has been shown to be the most likely modern digital tool to support economic development in developing nations [16]. 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 [1], if a mobile phone is shared by a group of people, it can be afforded by even the poorest communities.

So far mobile phones have primarily only been used for communications in developing countries. Recent improvements in mobile phone functionality and processing power make it a compelling device for piggy-backing other kinds of information services to areas thus far untouched by information technology.

UIST'05, October 23-27, 2005, Seattle, Washington, USA.

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

THE CAM USER INTERFACE

In contrast to other tangible UI programming toolkits [2, 9], CAM is not an API for software developers to build different physical user interfaces and applications. CAM is more like the World Wide Web, in that it 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 using a common infrastructure.

Since CAM uses standard Internet transport protocols for delivery (HTTP and SMTP), we focus on the user interface and the programming language that drives the user interaction.

CamBrowser

The CAM client application is called the CamBrowser, and has currently been implemented for Nokia Series 60 camera phones. CamBrowser is designed to process specially designed CamForm documents (see Figure 2). CamForms contain visual codes - two-dimensional data glyphs containing up to 76-bits of data that can be decoded from a camera image [18]. Visual codes serve as references to the interactive content embedded within CamForms.

Other researchers have presented a rich set of interactions made possible by tilting, rotating and translating the phone's camera relative to an individual visual code [19]. However it also has been observed that most of these interactions are not accessible to novice users [22]. In CAM we have simplified the interaction to two primitives. The user can *scan* codes from low-resolution images taken in real time, or *click* codes by taking a high-resolution image using the joystick button. Our focus was on designing a simple and intuitive interaction model with well-defined affordances between actions.

CamShell

CamShell is the programming language that is used to define interaction with visual codes embedded in the form. There is one visual code in every CamForm that serves as a *form identifier*. This code has a 0 as the lowest-order bit. The next 4 bits identify the form to the CamBrowser application. This information is used to load the *form description schema*, an XML file containing metadata about the form and the underlying data elements (including any default or pre-existing values). The remaining bits specify the protocol that should be used to load the schema file (currently HTTP, bluetooth or the local filesystem), and the required information to find the schema (a network address or a file name).

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. The XML elements contain the executable code that should be invoked when the callback is activated. The executable code itself is written using a simple scripted programming language including support for function calls, control flow, arithmetic and basic datatypes [21].

The mapping to the appropriate callback is provided by the next 7 bits of the action code. Static data is contained in the remaining bits, which is sent to the callback function as a parameter. There is a default function that is called when a form is first loaded. This is used to perform any required initial processing on the form.

CamShell provides an API for accessing the mobile phone's functionality - including the phone's user interface, networking and telephony features. Listed below are the main functions currently included in this API:

- **User Dialogs** - These instructions launch various user dialogs to collect data, get confirmation or convey a message to the user. Each dialog can be associated with audio and graphical prompts. Audio feedback has been found very useful by potential users.
- **HTTP Post, Get** - These instructions are used to make HTTP requests using the phone's built-in connectivity.
- **SMS, MMS, Email** - These instructions transmit asynchronous network messages. In the case of MMS and Email the current form image can be attached.
- **Phone Call** - This instruction is used to make a phone call to a particular number.
- **Applications** - These instructions are used to launch other applications, such as the Web or WAP browser.

Each callback function can contain any number of sequential actions, some of which may be 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>
```

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.

An example CamForm is shown in Figure 2. This CamForm is organized into the following three sections:

- **Header** - 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. A default form action can also be included in the header that triggers all of the required data entry tasks. This is useful for forms that have a single execution path.

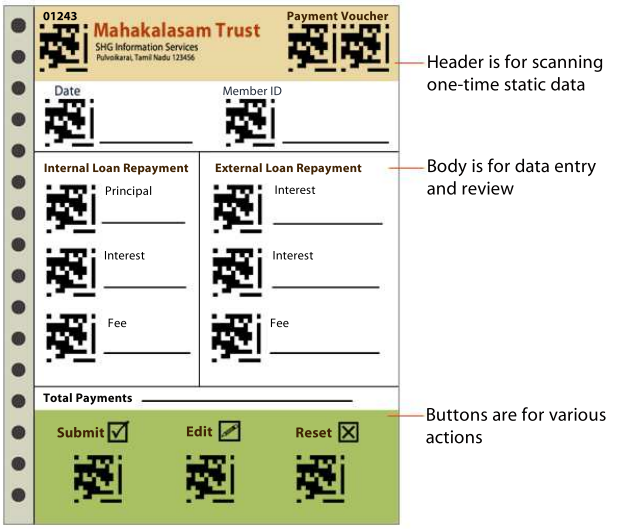

Figure 2: An example CamForm, showing the three sections the user can interact with.

- **Body** - 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 of the field.
- **Buttons** - 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.

USABILITY EVALUATION

We conducted an initial evaluation of a prior version of this system [14] with microfinance staff and group members in rural Tamil Nadu. This helped us identify the basic usability issues, many of which have been rectified in the current design. The users' response to the system was very positive, particularly when compared to our earlier experience designing a PC interface for a similar user population [15].

We are now conducting a more detailed evaluation for simple data entry tasks using both CAM and a web-based interface on a PC. We are making this comparison because the alternative to CAM is that data is collected on paper forms in the field and entered using a PC at the branch office. The field staff waste time in travelling to the branch office for this task. This reduces the number of clients served by each staff (a benchmark for efficiency in the microfinance industry), and takes time away from other tasks such as recruiting new clients and developing the capacities of existing ones. There is also a delay between when the data is collected and when it is available for reference.

A mobile interface is preferred because it would allow field staff to enter data as it is generated in the context of their regular documentation activities. Clients would also gain confidence by observing this process, particularly when hearing the audio confirmation that their transaction has been recorded. Earlier efforts to automate microfinance data collection using handheld devices faced problems in efficiency and cost. If we can demonstrate that CAM has comparable efficiency, accuracy and learnability to a PC-based system, without a significant increase in overall cost, it could have important ramifications for the microfinance industry.

Current mobile data entry interfaces are known to be inefficient and difficult to learn. Much of this difficulty is due to the limited screen space of mobile devices, which makes navigation between different tasks and data fields difficult. Our system addresses this issue by expanding navigation to the physical domain of paper documents.

APPLICATIONS

We hypothesize that these benefits can also carry over to other similar paper-based application domains in the developing world. We have met with banks, private companies, NGOs and governmental agencies that have expressed interest in using the CAM framework. Listed below are some of the other applications that we are exploring:

- **Sales and Distribution Tracking** - CAM can be used to track sales and distribution of inventory over a wide area.
- **Mobile Cash Register** - CAM can be used as an improvised cash register for rural provision shops.
- **Health Data Collection** - CAM can be used to collect health information from underserved rural populations. This data could be useful for primary health clinics, NGOs and government agencies.
- **Transaction Processing** - CAM can be used to securely process remote financial transactions. This would allow for the establishment of low-cost human-mediated banking proxies. Several Indian banks are exploring this method of service delivery using alternate technologies, including PC kiosks and smartcards [13].
- **Rural GIS** - CAM can be used in conjunction with a location-sensing technology such as GPS or PlaceLab [10] to populate Geographic Information Systems, useful for the government and other civil service organizations.

RELATED WORK

Other researchers have commented on the importance of paper in workplace settings [20, 12], and sought to improve coordination between paper and digital media [6, 4, 8, 3]. Companies have extended these efforts to develop commercial document processing systems that recognize structure and text from scanned paper forms and integrate with back end data sources [5]. None of these systems have used a mobile phone as the primary image capture and data entry device, and none provide a general purpose networked programming environment for specifying user interaction and client processing.

The Mobile Server Toolkit described by Toye et al. [22] is a visual-tag based system that allows mobile devices to access site-specific information services. The MST transmits individual UI events directly to an application running on

a bluetooth-connected server. CAM handles all user interaction on the phone itself - messages to the server contain data elements only. CAM can also work offline and send messages asynchronously using standard Internet protocols (HTTP and SMTP). These features make it more suitable for accessing remote information services without access to other nearby devices and only limited connectivity.

The Cooltown project [7] at HP Labs uses RFID, barcodes and and other sensing technologies to retrieve specific HTML documents. The visual code widget library developed by Rohs [17] includes a declarative UI specification that links visual codes with a rich set of data entry widgets. Neither of these systems support narrative interfaces consisting of sequences of actions and conditional execution. Mobile UI designers have recognized that sequential execution is more suitable for devices with limited screen space [11].

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 our observations into a usable and locally harmonious system that we believe is well-suited for an important set of users and applications in the developing world.

REFERENCES

1. Grameen Phone home page. http://www.grameenphone.com, March 2005.
2. S. Greenberg and C. Fitchett. Phidgets: easy development of physical interfaces through physical widgets. In *UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology*, pages 209–218, New York, NY, USA, 2001. ACM Press.
3. F. Guimbretière. Paper augmented digital documents. In *UIST '03: Proceedings of the 16th annual ACM symposium on User interface software and technology*, pages 51–60, New York, NY, USA, 2003. ACM Press.
4. J. M. Heiner, S. E. Hudson, and K. Tanaka. Linking and messaging from real paper in the paper pda. In *UIST '99: Proceedings of the 12th annual ACM symposium on User interface software and technology*, pages 179–186, New York, NY, USA, 1999. ACM Press.
5. IFP success stories, June 2003. http://www2.clearlake.ibm.com/GOV/ifp/success_stories.html.
6. W. Johnson, H. Jellinek, J. Leigh Klotz, R. Rao, and S. K. Card. Bridging the paper and electronic worlds: the paper user interface. In *CHI '93: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 507–512, New York, NY, USA, 1993. ACM Press.
7. T. Kindberg. Implementing physical hyperlinks using ubiquitous identifier resolution. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 191–199, New York, NY, USA, 2002. ACM Press.
8. S. R. Klemmer, J. Graham, G. J. Wolff, and J. A. Landay. Books with voices: paper transcripts as a physical interface to oral histories. In *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 89–96, New York, NY, USA, 2003. ACM Press.
9. S. R. Klemmer, J. Li, J. Lin, and J. A. Landay. Papiermache: toolkit support for tangible input. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 399–406, New York, NY, USA, 2004. ACM Press.
10. A. LaMarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes, F. Potter, J. Tabert, P. Powledge, G. Borriello, and B. Schilit. Place lab: Device positioning using radio beacons in the wild. In *Proceedings of PERVASIVE 2005, Third International Conference on Pervasive Computing*, Munich, Germany, 2005.
11. C. Lindholm, T. Keinonen, and H. Kiljander. *Mobile Usability: How Nokia Changed the Face of the Mobile Phone*. McGraw-Hill, NY, NY, USA, 2003.
12. W. E. MacKay. Is paper safer? the role of paper flight strips in air traffic control. *ACM Trans. Comput.-Hum. Interact.*, 6(4):311–340, 1999.
13. T. S. Parikh. Rural microfinance service delivery: Gaps, inefficiencies and emerging solutions. http://www.cs.washington.edu/homes/tapan/papers/mf-challenges.pdf, April 2004.
14. T. S. Parikh. Using mobile phones for secure, distributed document processing in the developing world. *IEEE Pervasive Computing*, 4(2):74–81, April 2005.
15. T. S. Parikh, K. Ghosh, and A. Chavan. Design studies for a financial management system for micro-credit groups in rural india. In *CUU '03: Proceedings of the 2003 conference on Universal usability*, pages 15–22, New York, NY, USA, 2003. ACM Press.
16. The real digital divide. *The Economist*, Mar. 2005.
17. M. Rohs. Visual code widgets for marker-based interaction. In *IWSAWC'05: Proceedings of the 25th IEEE International Conference on Distributed Computing Systems – Workshops (ICDCS 2005 Workshops)*, Columbus, Ohio, USA, June 2005.
18. M. Rohs and B. Gfeller. Using camera-equipped mobile phones for interacting with real-world objects. In A. Ferscha, H. Hoertner, and G. Kotsis, editors, *Advances in Pervasive Computing*, pages 265–271. Austrian Computing Society, Vienna, Austria, 2004.
19. M. Rohs and P. Zweifel. A conceptual framework for camera phone-based interaction techniques. In *Pervasive 2005, Third International Conference, Lecture Notes in Computer Science (LNCS) No. 3468*, Munich, Germany, May 2005.
20. A. J. Sellen and R. H. Harper. *The Myth of the Paperless Office*. MIT Press, Cambridge, MA, USA, 2003.
21. Simkin language, May 2005. http://www.simkin.co.uk/.
22. E. Toye, R. Sharp, A. Madhavapeddy, and D. Scott. Using smart phones to access site-specific services. *IEEE Pervasive Computing*, 4(2):60–66, April 2005.