Designing an Architecture for Delivering Mobile Information Services to the Rural Developing World

Tapan S. Parikh and Edward D. Lazowska
Department of Computer Science and Engineering
University of Washington
tapan,lazowska@cs.washington.edu

ABSTRACT

Implementing successful rural computing applications requires addressing a number of significant challenges. Recent advances in mobile phone computing capabilities make this device a likely candidate to address the client hardware constraints. Long battery life, wireless connectivity, solid-state memory, low price and immediate utility all make it better suited to rural conditions than a PC. However, current mobile software platforms are not as appropriate. Web-based mobile applications are hard to use, do not take advantage of the mobile phone's media capabilities and require an online connection. Custom mobile applications are difficult to develop and distribute. To address these limitations we present CAM - a new framework for developing and deploying mobile computing applications in the rural developing world. CAM applications are accessed by capturing barcodes using the mobile phone camera, or entering numeric strings with the keypad. Supporting minimal navigation, direct linkage to paper practices and offline multi-media interaction, CAM is uniquely adapted to rural device, user and infrastructure constraints. To illustrate the breadth of the framework, we list a number of CAM-based applications that we have implemented or are planning. These include processing micro-finance loans, facilitating rural supply chains, documenting grassroots innovation and accessing electronic medical histories.

Categories and Subject Descriptors

H4.3 [**Information Systems Applications**]: Communications Applications; H5.2 [**Information Interfaces and Presentation (e.g., HCI)**]: User Interfaces; C.2.4 [**Distributed Systems**]: Client-server, distributed applications

General Terms

Design, Human Factors

Keywords

mobile computing, mobile phones, paper user interface, rural development, ICT, client-server distributed systems

1. INTRODUCTION

Providing timely and efficient information services in rural areas of the developing world is a difficult task. Current information practices are overwhelmingly paper and memory-based. This imposes severe limitations on the aggregation and dissemination of information. However, due to environmental and user constraints, the introduction of computing to automate these processes is equally challenging.

Most rural villages in the developing world do not have the economy or infrastructure required to support a computer. Therefore, rural people must travel to larger towns or cities to access digital resources, either in a public place or via an intermediary. Given the condition of rural roads, and the inconsistency of public transportation, this requires significant time and motivation.[1] Otherwise they must wait for someone to bring resources to them in a medium that they can access. In either case, the latency is high and the access is limited.

However, there is a revolution in the making. Just as personal computers changed the nature of information access and communications in the developed world in the 1990s, today *mobile phones* are changing their nature worldwide [29]. The lower cost of wireless infrastructure, deregulation in the telecommunications industry and the plummeting cost of handsets is putting mobile telephony in the hands of billions of people around the world.

The newest mobile handsets developed by several manufacturers provide an open application development platform and significant computing capabilities [37]. Given the popularity of these devices with developing world populations, and the immediate utility of voice communications, these 'smartphones' are an opportunity to bootstrap computing in the developing world. They are small, handheld computers with some constraints and additional features. Several of these (battery operation, solid-state memory, wireless connectivity, affordable price) could make them a better suited device for rural developing world conditions than a conventional PC.

However, current mobile software platforms are notoriously difficult to use and develop for, and make the assumption that there will be ubiquitous connectivity. To foster an information revolution in the rural developing world, each

[1]However, this is not unheard of. The first author knows an organic farmer in Gujarat, India who regularly travels eight hours by bus to carry on an e-mail correspondence with a like-minded colleague in Europe. This colleague once sent the farmer a computer, but he still couldn't access the Internet because he didn't have a phone line.

of these problems must be rectified. Applications must be used by minimally educated users, developed by minimally trained developers and meant to work in a variety of connectivity and power environments. To address these requirements, we present CAM - a new framework for developing and deploying mobile applications in the rural developing world. Supporting minimal, paper-based navigation, a simple scripted programming model and offline interaction, CAM provides a platform uniquely adapted to rural computing requirements.

In prior work, we introduced the idea of processing paper documents with a camera-equipped mobile phone [23], and formally evaluated the usability of a prototype system with rural users [25]. Here we extend this work in several specific ways. We provide a detailed description of the CAM system architecture, including a new numeric navigation interface and asynchronous networking approach. We discuss the specific characteristics of the rural developing world environment that have driven the design choices we have made. We outline the advantages of our system compared to other platforms for rural computing. To illustrate the breadth of the framework, we list a number of CAM-based applications that we have implemented or are considering. We believe this description will be useful for other researchers developing rural computing systems, or are interested in using CAM for their own applications.

2. CHALLENGES

Rural areas in the developing world pose a number of challenges for deploying computing technologies. In this section we discuss some of these challenges. While most of these observations were drawn from the author's experiences in rural India, they are indicative of similar conditions existing in other developing regions around the world. An appreciation of these challenges is crucial to designing a suitable system architecture.

2.1 Environmental Challenges

2.1.1 Intermittent Power

The power grid in the developing world is notoriously unpredictable. Both spikes and outages are common. There is *load-shedding* in both cities and rural areas (scheduled power outages to conserve energy). While the authors have often enjoyed power outages as an opportunity for a restful nap or a stroll along a village road, the repercussions are not always so pleasant. A trail of fried laptop batteries and hard disks attests to this fact. A battery-powered uninterruptible power supply (UPS) is a necessity for any PC in the developing world. This creates an additional expense that consumers are reluctant to bear. Even for laptops and mobile phones, a surge protector helps to maintain the life of the battery and AC power supply.

2.1.2 Intermittent Connectivity

As opposed to the temporally intermittent nature of power, connectivity is *spatially intermittent*. Landlines are prohibitively expensive to extend to villages with a limited revenue base. Even if a village is connected, it is probably only with low-quality copper wire. The local phone exchanges may use outdated equipment that hinders digital communications. Internet connections established over these links are slow and frequently disconnected. Cellular penetration is growing, but thus far wireless towers are concentrated near cities, large towns and major roads. It is possible that novel wireless technologies will eventually make connectivity more accessible, but differing topographies and population densities means that some percentage of the world's population will never be within arm's reach of a connected device.

2.1.3 Long Travel Times

The quality of roads is generally poor in rural areas, increasing driving times considerably. Public transportation, while ubiquitous in many countries, is also delay-prone. Several transfers may be required to travel from one place to another. Travel is incredibly time-consuming, often taking an order of magnitude more time than in the developed world.

2.1.4 Variable Population Density

One factor that varies greatly from country to country and continent to continent is population density. To note two extremes, in Bangladesh the density is over 1000 people per square kilometer, while in some parts of Central and Southern Africa it is less than 10 [28]. This greatly impacts the economics of providing information access - in countries with less density the same physical infrastructure will serve fewer people, and have a correspondingly smaller revenue base. Infrastructure technologies must be cheap and flexible enough to serve all possible conditions, or different solutions will be required for different areas.

2.1.5 Lack of Secure Storage

In the developed world we are accustomed to having an office, home or apartment that we can lock to secure expensive devices. In contrast, a typical one or two-room dwelling in a village can be shared by many family members. The notion of a locked door is virtually unheard of. The housing materials can be penetrated, especially in the tropics. The only security against theft is that someone will observe the action, or that the perpetrator will be identified through the village social network. These are both usually effective safeguards. However, it is virtually impossible to limit access from friends and relatives. The only protection against the environment is to store valuables in a more solid structure, like the village school, a business or government office. However, these structures may be dominated by the village oligarchy which can restrict access, possibly excluding lower classes and/or women. The best place for valuables is a sturdy safe.

2.2 User Challenges

2.2.1 Limited Education

The low quality of schools in rural areas, and the demands of agricultural work, often force children to abandon schooling at an early age. Those that do succeed in attaining a high school or college education seek employment in the city, where salaries are considerably higher. Less educated people have difficulty with abstraction and symbolic manipulation. Many rural people, especially those in older generations, may be illiterate. If they are literate, it is almost definitely in the vernacular.

However, predicting that computing technologies will directly be used by illiterate or semi-literate people is sometimes a naive assumption. Many villages have an educated, literate member that they rely on for performing tasks beyond their individual capacity. At the same time, it is important that the illiterate person *understand* the process, so that they can safeguard their personal interests (for example, when performing financial transactions).

2.2.2 Underemployment

The lack of economic opportunity in villages, combined with the inconsistent performance of farming, creates chronic underemployment in rural areas. In large families, only the oldest sons may be accommodated in the traditional business. The rest must seek their own employment opportunities. Moreover, while farming is hard work, it is usually limited to certain seasons and times of the day. The rest of the day is spent idling in the village, at the local *paan* shop.

This idle workforce has been utilized by companies, banks, non-governmental organizations (NGOs) and political organizations working in rural areas. By hiring young, high-school educated *field agents* to travel from village to village, they can build an affordable and effective service delivery channel in rural areas. Sometimes paid on commission, field agents are benefited by knowledge of the local people and culture, and are accustomed to rural travel and living conditions. This model is common in microfinance, microinsurance, primary health care, and in retail supply chains, among other sectors.

2.2.3 Limited Disposable Income

Almost all of poor people's income goes towards their livelihood and social requirements [33]. Very little money is left for leisure or exploratory purchases. Given the dramatic difference in income and expenditure levels between cities and villages, electronic devices must be shared to even be within reach for most rural populations. A new system or technology must fulfill an immediately perceived need to be relevant in this context.

2.3 Things to Avoid

Given our experience developing rural computing applications, we list several features that pose particular difficulty for this domain.

2.3.1 Text

Text is not only a problem because of limited literacy. Hardware and software constraints make entry, storage and display of local-language text difficult. Keyboards are designed for English. Keyboard mappings for other languages are difficult to standardize or remember. Unicode character representations are Western-centric, and do not always adequately capture the intricacies of other scripts. Rendering complex scripts requires support at the operating system level. Fonts can be expensive, difficult to find, and based on non-standard encodings. It is difficult to manage the consistency and quality of translations across unfamiliar languages. Meeting these requirements dramatically increases the technical complexity of a development project.

2.3.2 Abstract Navigation

Users with a limited education have difficulty applying abstractions or understanding symbolic manipulation. Rural people are accustomed to the direct manipulation world of farming and labor. In our earlier experiments, we found that getting uneducated users to understand the notion of menus, screens, and windows was an altogether unsuccessful exercise [24]. While it is *possible* for users (especially the young and the high-school educated) to grasp WIMP (windows, icons, menus, pointer) concepts, it adds significantly to the training and documentation requirements of computing systems. For some users, no amount of experience will make them truly comfortable (as anyone with a computer-fearing parent or grandparent can attest).

2.3.3 Excessive Documentation

The primary medium of communication in rural villages is word of mouth. Any system that requires excessive documentation will limit its reach to those that can access and consume that knowledge. On the other hand, a system that is easy enough so that users can learn how to use it from one another will have a much wider reach. The best way to leverage the strength of rural social networks is by making the operation of the system conveyable by *word of mouth.*

2.3.4 Personal Devices

A personal device is a completely foreign concept for developing world cultures and economies. Mobile phones, computers, televisions, radios, etc. are almost always shared by many people from the family and community. Not only is this often the only way for communities to afford these devices, but Western notions of privacy and personal property have very different applications in rural villages.

2.3.5 Online Interaction

As discussed earlier, rural areas have spatially intermittent connectivity. By requiring online usage of an application, villagers would have to travel to a connected location to access it. As discussed earlier, this can be very inconvenient, possibly taking an entire day.

3. LIMITATIONS OF CURRENT MOBILE APPLICATION PLATFORMS

Mobile phones seem like an excellent way to introduce information technology to the rural developing world [29]. Their low cost, long battery life and immediate utility directly address several of these challenges. Moreover, because they are portable, they can be carried back and forth between connected and disconnected regions, ferrying data back and forth [27]. While still requiring travel, now one device and one person can perform this task on behalf of a village or region. Field agents carrying mobile devices can offer a variety of services to rural residents. Wireless infrastructure exists in almost every country, making this approach flexible enough to deploy anywhere.

However, while mobile phone hardware is well-suited to rural conditions, the same can not be said about software. Mobile web interfaces are notoriously hard to use, even for developed world users. Typing URLs with a numeric keypad is slow and painful. Therefore, users must rely on a portal or set of bookmarks to access web sites. Most web pages are designed for larger screens, making navigation within a page also problematic. Interaction is based on a constrained rendition of the WIMP metaphor that is awkward at best. Mobile phones also do not provide any way to work with web content without an active connection. Moreover, most web-based applications do not take advantage of the built-in features of mobile phones, like a microphone, speakers or even a camera.

Developing custom applications for mobile phones is becoming more common. However, this requires knowledge of new APIs (some of which require expensive licensing). Distributing applications is cumbersome. Either providers must push content to users, or users must navigate to web sites and download the software. Due to the difficulty of using mobile web browsers, the easiest way is often to use a PC to download the installer and then transfer it to the phone using bluetooth or USB.

4. THE CAM ARCHITECTURE

In this section we present CAM, a mobile phone application platform that addresses these software limitations. By supporting minimal, paper-based navigation, a simple scripted programming model and off-line multi-media interaction, CAM provides a platform uniquely adapted to rural computing requirements.

4.1 System Overview

The driving element of the CAM architecture is a mobile phone application called the *CAMBrowser*. CAMBrowser has been implemented for several Nokia phone models based on the Series 60 platform [37]. (We intend to port this application to other mobile platforms in the future.)

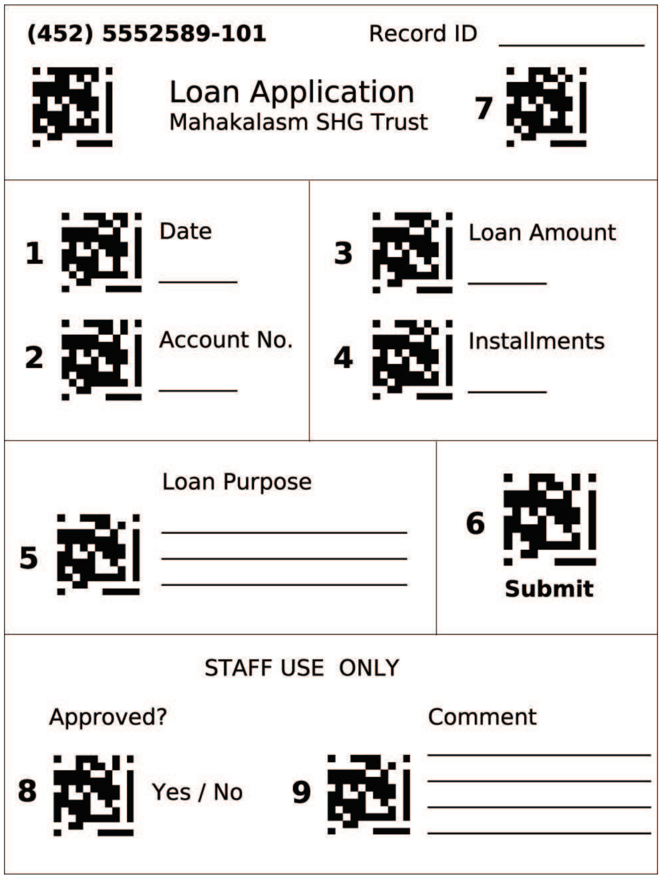
Users navigate within and between CAM applications by capturing barcodes using the mobile phone's built-in camera, or by entering numeric strings. These barcodes and numbers can be printed directly on paper forms for ready access (see Figure 1). Forms-based data entry is extremely common in the developing world. *CAMForm* analogs of existing paper forms serve as efficient offline clients for CAM applications. Data is first entered on paper, from where it is transcribed, processed and uploaded using the CAMBrowser. Data transfer can happen either immediately, or later when the phone has a connection.

The CAMBrowser downloads and executes applications written in Simkin, an XML-based scripting language including support for function calls, control flow, arithmetic and basic datatypes [38]. CAM provides an API for accessing the mobile phone's user interface, networking and multimedia input and output capabilities. Applications are downloaded on demand from an online source, either via the web or a multimedia message (MMS). The XML is cached locally on the phone along with its associated binary data, in a directory structure organized according to the server and application ID.

In the following sections we describe the details of the CAM architecture, including its navigation, content development and networking features, and discuss how each of these addresses the rural developing world's computing requirements.

4.2 Navigation

In an earlier design experiment, we observed the navigation difficulties encountered by semi-literate rural users when using a traditional PC interface [24]. These problems can be exacerbated by the limited display and input capabilities of a mobile phone. Even if users are literate, traditional menu-based navigation is complicated, especially on a mobile phone, requiring significant time to understand and convey. Moreover, in a menu-based system, only a limited number of options can be accessed at one time. Other researchers have noted the difficulty of hierarchical navigation


(452) 5552589-101 Record ID

Loan Application
Mahakalasm SHG Trust
7

1 Date
3 Loan Amount
2 Account No.
4 Installments

5 Loan Purpose
6 Submit

STAFF USE ONLY
Approved?
Comment
8 Yes / No
9

Figure 1: An example CAMForm, for submitting a microfinance loan application.

for rural users [3]. Entering a URL in English using a numeric keypad is completely unrealistic for these users.

To circumvent the limitations of standard navigation methods, we have taken a different approach. Applications and application functions are indexed using numeric strings, encoded either as barcodes to be captured via the mobile phone camera, or as numbers to be entered via the keypad. Both barcodes and numbers can be printed on paper forms. In this way navigation is directly tied to a paper representation of the task. The paper form also serves as a local record of the action if a printer is not available.

For barcode recognition and generation, we are using the open source toolkit developed by Rohs and Gfeller [32]. This includes a 2D barcode recognition library for Nokia Series 60 phones. A user can capture a barcode by taking a picture of it using the mobile phone camera. Each barcode contains 83 bits of data, 7 of which are being used for error correction (leaving 76 bits of effective data carrying capacity). Several barcodes can be detected from a single camera image.

To navigate to a new application, the user captures a barcode in the top-left corner of a CAMForm (see Figure 1). This barcode has its first two bits set to 0. The next two bits represent the protocol to be used for downloading the code and data (currently either HTTP or SMS), and the next 48 bits encode the address (either an IP address or an SMS phone number). The last 24 bits specify a server-specific application index. Application code and data is requested from the server and cached locally on the phone for offline access (this will be discussed in the networking section).

The same application can be accessed by entering a numeric string using the phone keypad. This is useful for mobile phones without a camera, or for applications that do not have an associated paper representation. The numeric string can be structured like a phone number, providing a familiar metaphor for rural users. An application is "dialed" by first entering an arbitrary length phone number (or 12-digit IP address) for the server, a '*' delimiter (two '*'s for a IP address), and then a server-specific application ID. If the address or phone number is omitted, it is treated like a local application (the equivalent of accessing localhost in a web browser).

4.3 Content

Users navigate within an application in a similar way. The functions exported by each application are specified as elements in an XML file, written in the Simkin scripting language. Barcodes with the first two bits set to 1 refer to application functions. The next 8 bits specify an application-specific function ID (each application can export up to 256 unique functions). The rest of the bits in the barcode are sent to the function as a static parameter.

Using the keypad, functions are accessed by entering just the numeric function ID. Parameters can be specified using an optional '#' delimiter. This string can be concatenated at the end of the application string described in the previous section, separated by another '#' delimiter. This creates a unique *numeric URL* that can be used to remember and share references to arbitrary application functions. Table 1 details the barcode and numeric keypad inputs used to access various CAM applications and functions.

If an application is linked to a forms-based process, the barcodes and IDs for functions can also be printed on forms for contextual access. These *CamForms* are designed using standard word processing software. Barcodes are generated using a PHP script, and pasted into the forms manually. We are currently designing an IDE that will support integrated authoring of CAM forms and applications.

Figure 1 shows a CAM-enabled loan application designed for a microfinance institution. The barcode in the top right, when clicked, activates a function that displays a sequence of prompts for the user to enter each of the values from the form. A section of the code for this function is shown below. *input_date*, *conf_note*, *recordaudio* and *put* are all CAM API functions. A full list of API functions is given in Table 2.

```
<function name="7_click" params="param1">
if (!input_date(date, "dprompt.wav", "dprompt.bmp"))
    return false;
...
if (!conf_note("purpprompt.wav", "purpprompt.bmp"))
    return false;
recordaudio("data\".recordid."\purp.wav");
if (!conf_note("cprompt.wav", "cprompt.bmp"))
    return false;
put("mms","+12065551695","data\".recordid."\");
</function>
```

The prompts are displayed in sequence rather than laid out spatially (like in a web-based entry form). This is better suited to the small screen of the mobile device, on which it is difficult to display or enter several pieces of information at one time. We call this "wizard" interaction, as it resembles the task wizards used for installing or configuring software applications. Users are guided step-wise through the process, reducing navigation requirements. This method avoids "decision points", where the user has to cognitively pause after each action to decide what to do next, and how to do it. This pause can create unnecessary confusion for novice users. Instead, in our system, users just choose a high-level task and then follow the prompts.

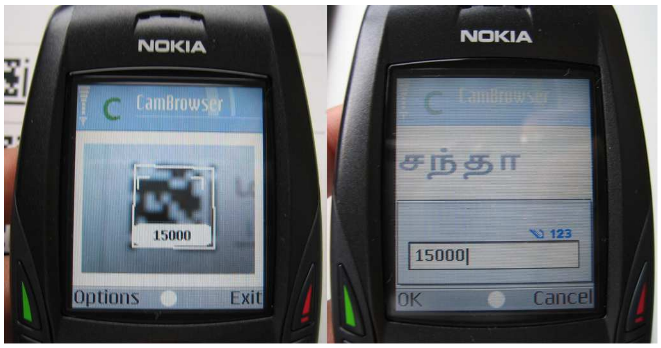


Figure 2: On the left, reviewing an entered value by focusing on a barcode. On the right, clicking brings up a prompt for editing the value. An audio prompt is also played.

Figure 2 shows a CAM prompt in the regional Tamil language. However, the mobile phone operating system does not support Tamil data entry or display. The text is displayed as a bitmapped image, accompanied by a voice clip indicating the name of the field. Every CAM prompt can be associated with arbitrary audio and graphics. This increases the flexibility of the system, especially for dealing with unsupported languages or illiterate users. Even if someone else is performing the task, an illiterate user can listen and hear the prompts as data is being entered. The audio feedback contributes to the wizard metaphor, making the interaction proceed like a conversation between the user and the device. The device asks a question, and the user answers. When we tested a CAM application in rural India, some users didn't even look at the screen prompt before entering a value [25].

To address the problem of local language textual input, we took a different approach. Most of the values to be entered from the form in Figure 1 are numeric values. Instead of asking for the loan applicant's name or login, we rely on a numeric account number. The loan purpose is captured by recording an audio clip. We could have also prompted the user to capture an image of the entire form, or a specific entry field. If required, voice and image data can be transcribed later at the bank office, where a PC is available,

Application	
Barcode	00-Protocol(3 bits)-Addr(38)-AppID(24)
Numeric	[Addr-*]-AppID-[FuncID[#Param]]
Function	
Barcode	11-FuncID(8 bits)-Param(66)
Numeric	FuncID[#Param]

Table 1: Barcode and numeric inputs for accessing CAM applications and functions.

input_int, *input_date*, *input_password*, *input_pin*, *input_phone_number*, *input_time*, *input_number*, *input_text*	Prompts asking the user to enter different kinds of values
record_audio, *take_picture*	Ask the user to record an audio clip, or take a picture of a form or other object
message_note, *conf_note*	Send a message to the user, or display a confirmation dialog
get, *put*	Retrieve or submit application data to the server using using http, sms or mms
encrypt, *decrypt*	Encrypt or decrypt application data with a specified key
sms, *mms*, *email*	Send arbitrary messages
phonecall, *browser*	Make a phone call or launch a web browser
log	Write a string to the application log

Table 2: Functions provided by the CAM API. All prompts and messages can be accompanied by arbitrary audio and/or images.

to be performed by staff trained in local language keyboard input. We can refer back to these transcribed records using a numeric identifier, like we did with the account number.

After data has been entered, the user can review the entered values before submitting them to the server. By focusing the mobile phone camera on a barcode associated with a form field, the current value is displayed on the screen, next to the value entered on paper (see Figure 2). By focusing on the loan purpose field, the recorded audio clip is played. If any of the values is incorrect, the user can capture an image of that barcode, displaying a prompt to edit the value. The implementation is the same as that for other functions. Each barcode is linked to separate *scan* and *click* callback functions. When scanned, the value returned by the scan callback is displayed. This is the value of the field. When clicked, a prompt is displayed to edit the value. Example scan and click callback functions are shown below.

```
<function name="1_scan">
return date;
</function>

<function name="1_click">
input_date(date, "dprompt.wav", "dprompt.bmp");
</function>
```

4.4 Networking

The first time CAMBrowser encounters an application, it attempts to download the code and data from the specified server. The barcode or numeric string specifies the protocol to submit the request. Currently, the options are sending a HTTP request to a web server, or a SMS to a phone number. In the first case, the response is received immediately as XML over HTTP. In the second case, the application code and data is sent by the server as a MMS message with a SIS attachment. A SIS file is a self-extracting installer for the Series 60 platform.

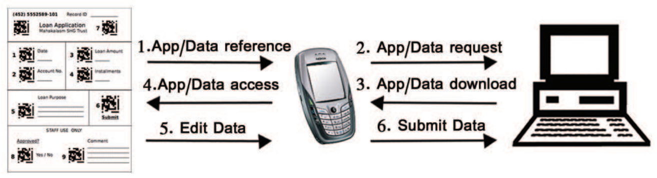


Figure 3: An example set of steps in accessing and using a CAM application. Except for steps 4-5, there can be arbitrary time and distance between each step, supporting an offline, asynchronous interaction model appropriate for rural areas.

The SMS-based method supports offline use. The first time a disconnected phone tries to access an application, it might not be available. As soon as the browser sees the form, it creates a SMS request, which is cached in the phone's outgoing message queue. The message will automatically be sent when the phone enters a connected area. After the SMS is received by the server, it sends back the appropriate SIS file to the sender as a message attachment. The phone will automatically download the message if it is connected. When the user opens the message, they will be guided to install the application in the proper location. The complete code for the application is then cached on the phone for offline use. The next time the phone returns to the village, the application can be used.

Eventually some applications may have to be flushed from the cache to make room for others. By implementing an intelligent caching scheme, we can ensure that the most frequently used applications are retained. This has not been a requirement thus far due to the small number of CAM applications that have been developed. All of these easily fit within the available flash memory of the phone (32 MB, expandable to 1 GB). The user can also manually request a flush of the local cache, or force a server refresh of a particular application. The CAM API supports sending a refresh request to the server programmatically.

Application data is also cached locally. Captured data values are stored as XML, while recorded audio and images are stored as binary files. This data is stored in a specific directory for each application. Multiple records can be cached by further sub-dividing this directory using a unique numeric ID for each record instance. This record ID can be printed on a form (either as a numeric string or barcode), or automatically generated and displayed when first saving the data. The user can copy the generated ID back to the form for future reference (for example, in the top right of Figure 1). Later, the user can review or edit previously entered records by entering the right ID or barcode.

The phone's memory serves as a local cache of the data stored on the server. The application can programmatically copy the data for any directory from the phone to the server, and vice versa. This is accomplished using the *get* and *put* functions. For a *put*, the application sends data either as a HTTP POST request, or as an attachment to a MMS message. Once again, the MMS message is cached in the outgoing queue until a connection is available. The application is responsible for choosing the right protocol based on its requirements, the configuration of the server and the available connections.

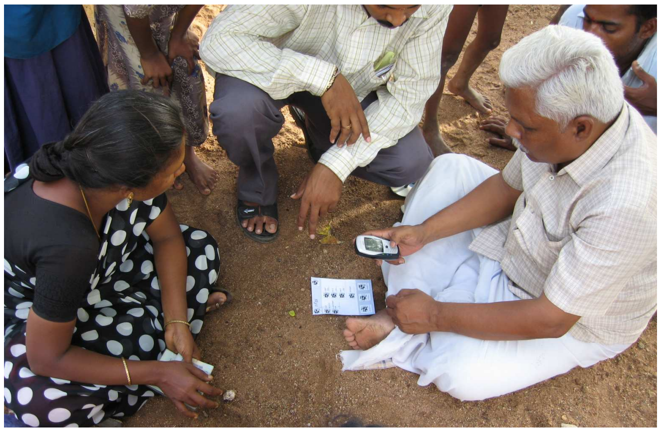


Figure 4: A bank field agent processing loan applications using the CAMBrowser during our initial usability trials.

The *get* function works in a similar way, using either HTTP or SMS. In the former case, a HTTP GET request is sent to the specified server, with an optional set of parameters. The response is sent as a XML message. Again, SMS can work in an offline fashion. The request is cached in the outgoing message queue until the phone is connected. The response is sent as a SIS file attached to an MMS message. If a *get* is issued in an offline environment, the requested data will be downloaded and cached by the next time the phone returns to the village.

Data can be encrypted using an encryption key printed as a barcode. This barcode remains with the end user, precluding others from accessing the data without it. Later, when the user wants to access the data again, he or she can provide the associated decryption key to the CAMBrowser application. A password or PIN provides additional security.

Trying to access an application that is not cached on the phone generates an implicit *get* request for the application directory. Similarly, if the user is trying to access a record directory that is not cached, a *get* request for the record subdirectory can be issued. In this way, users can request data for a specific set of records without the phone having any prior knowledge of the application. Given the high latency of the village roundtrip, it is important to capture as many data requests as possible up front.

The server-side functionality can be implemented in a variety of ways. It can be a standard web application backed up by a database, or it can just be a simple filestore. The server can receive and process SMS and MMS messages via an attached phone with its own phone number. The server-side application is responsible for maintaining coherency and ordering conflicting messages in a logical way.

5. AN EXAMPLE SCENARIO

To better illustrate the actual operation of a CAM application, we describe the microfinance loan application scenario in more detail. We are developing this system for deployment with a microfinance institution in Tamil Nadu, India.

Many banks and microfinance institutions in the developing world are interested in building relationships with rural microfinance groups. They hire field agents to interact with villagers, and assess lending opportunities. While in the field, agents distribute loan application forms to those people interested in obtaining a loan.

Prospective clients fill out these applications at their leisure. If they are illiterate, a literate friend or family member can help them. The loan application includes fields for the client's current account number, the desired loan amount, term, and the loan purpose. The loan purpose is a summary of the client's reason for requesting the loan (buy livestock, start a small business, pay for health bills, etc.).

The next time the field agent comes to the village, he transcribes the loan applications using his bank-issued mobile phone. First he captures the application ID, which loads the appropriate CAM application. It is likely that the application is already cached on his phone (if not, it will be the next time he comes). The agent then captures the barcode in the top-right to enter all of the data from the loan application. Guided by the prompts, he enters the account number, the desired loan amount, the loan term and captures an audio clip of the client recounting the intended purpose of the loan.

The client listens to the audio prompts to follow along in the process. If he wants to verify that the data has been entered correctly, he can ask the agent to focus the camera on each of the fields to display the entered values. Once they are both satisfied, the agent captures the 'Submit' barcode to generate a MMS message. The message contains a XML file with the entered data, and an audio recording of the loan purpose. This MMS file is cached in the outgoing message queue. It will be sent when the field agent travels on the highway (a connected area) on his way to the next village. The client keeps the paper form for his own records.

At the bank office, the manager reviews his incoming messages and sees the new loan applications. Based on the purpose of the loan, and the clients' past credit history, he decides which loans should be issued. He uses a web-based interface to enter the corresponding approvals in the database. These decisions are automatically packaged in a SIS file and sent to the field agent's phone as a MMS message.

When the agent opens the message, new XML files are placed in the directory corresponding to each form instance. By reviewing the 'Accepted' field on each form, both the field agent and the client know which have been approved. The clients whose loans have been approved are asked to come to the bank branch to collect their money. For those that were denied, the bank manager includes an audio or textual comment indicating the reason. A summary can be hand copied to the paper form for later reference. The whole loan application process has been accomplished without the field agent ever having to return to the distant bank office.

6. ADVANTAGES OF CAM APPLICATIONS

In this section we outline some of the advantages of the CAM platform for developing rural computing applications.

- **Easy to Use** - CAM interaction is based on simple primitives that do not require knowledge of extended metaphors. Learning to use an application is as easy as taking pictures of barcodes, or entering numeric strings, and then following the prompts. The limitations of the smaller display and a numeric keypad dovetail nicely with this approach. During usability testing of a CAM application with microfinance field agents, we found that all users were able to use the system effectively with a few minutes practice [25].

- **Easy to Document** - This understanding is easily conveyed from person to person, without training or user manuals. In our testing, users explained the system to each other without any intervention. Being able to convey knowledge of how to use the system by word of mouth will allow CAM applications to scale virally and increase their impact correspondingly.
- **Tied to Paper Forms** - By linking CAM interaction to forms, the process becomes more familiar to users accustomed to paper-based tasks. Preliminary data entry can be done completely offline, without access to a phone. Query results can be sent as printed, possibly interactive reports. Given that one phone is probably being used for many tasks in many villages, this maximizes the efficient use of available time and resources. When the phone is available, data entry and review is done in the direct context of the paper form, reducing errors and increasing the trust of village users.
- **Localized without OS support** - Using audio and graphical input and output, a paper-based UI and a numeric index to applications and data, CAM applications can be localized even without OS-level support for the target language. Given the tremendous variety of languages and scripts in the developing world, and the difficulty of developing applications for languages not supported by the OS, this removes a tremendous practical obstacle to application development.
- **Easy To Distribute** - Because most people will not have their own dedicated device, distributing electronic references to applications and data to individuals is not feasible. By referring to applications and data using paper forms and numeric strings, they can be distributed via both person-to-person and paper-based channels. These can be exchanged without any access to a device.
- **Can be used Offline** - CAM applications can be used without an active Internet connection. While this increases the latency of information exchange, it is still much better than having to make a trip to the city for each task individually. Users can also perform preliminary data entry on paper forms without having access to any technology, providing another level of offline access. This improves the efficiency and accessibility of the system, given the high person / device ratio,
- **Easy to Bootstrap** - Due to the demand for voice communications, a mobile phone is much easier to introduce to rural areas than a more expensive and less immediately useful PC. Service providers can generate an initial revenue stream by offering calling services [9]. Later, other applications can be bootstrapped as demand increases and network effects set in.

7. EXAMPLE APPLICATIONS

In this section we describe some applications of the CAM platform. The first three we are currently implementing and deploying in collaboration with our NGO partners. The other two have been discussed with possible collaborators.

- **Capturing Microfinance Transactions** - Once microfinance loans have been issued, cash transactions have to be recorded to monitor timely repayment. We have developed CAM-augmented receipts to be used by field agents when collecting repayments from clients. These transactions are aggregated in a centralized database, where they can be reviewed by bank staff to ensure the repayment of loans, and that field agents are doing their job effectively. After hearing about our trial deployment with one microfinance institution, several more have expressed interest in adopting the same model.
- **Ordering Groceries** - One of these microfinance institutions operates a community grocery store. The community pools money every few months to buy household necessities (rice, grains, oils, etc.). Buying items in bulk allows the local people to purchase at better prices and get better quality. Using CAM, we have developed an application for villagers to submit their monthly orders to the grocery electronically. The grocery manager is able to better plan for that month's purchasing and avoid excess inventory.
- **Supply Chain Tracking** - We have developed CAM-augmented forms and shipment labels to monitor the distribution of products in a rural supply chain. These are going to be tested by a company involved in the manufacture and sale of health products in rural areas. Using this data, the company will monitor inventory levels in rural warehouses, and assess the performance of their traveling salesmen (i.e. field agents).
- **Documenting Village Inventions** - The authors previously helped develop a PC-based application for documenting rural innovations [12]. Farmers or other rural people who had developed a new agricultural practice or other invention (farm implement, organic pesticide, livestock health treatment, etc.) used a PC to submit basic data about their creation. These submissions were viewed by other farmers, scientists and researchers, who could provide comments and feedback, after trying out the idea on their own. Field agents were employed to capture audio and video documenting the most promising inventions. Using CAM, we can provide a much cheaper and more scalable way of collecting and distributing this data.
- **Linking to Electronic Medical Records** - Poor patients often receive inconsistent medical care because different providers don't know about their existing conditions and treatment history. Using CAM, every patient can be linked to a electronic medical history that is updated on each visit. This record can be accessed either via a barcode printed on a document carried by the patient, or by using a patient-specific numeric ID.

8. RELATED WORK

The CAM system draws upon prior research in three areas: mobile user interfaces, paper user interfaces and store-and-forward networks based on transport of physical media.

8.1 Mobile UIs

Many researchers have looked at the problem of interacting with content given the limited display and input capability of mobile devices. Most of the early work in this area used a proxy server to optimize web pages for presentation on a mobile device [14, 8, 2, 7, 34]. Both these and later projects relied on spatial techniques like text summarization and fragmentation, scaling of images, non-speech audio, thumbnail and fisheye views to display and interact with desktop content on a smaller screen [5, 18].

The Satchel project used lightweight electronic tokens to work with and share documents, reducing bandwidth and storage requirements on the mobile device [19]. Pascoe et al. designed a PDA application for mobile fieldworkers studying giraffes in Kenya. They observed that displaying prompts in sequence made the interface easier to use if the user was moving or occupied by another task [26].

The Wireless Application Protocol (WAP) is a specification used to develop custom web content for mobile devices [35]. However, researchers have found WAP applications difficult to use, with the protocol even being referred to as a "a temporary aberration that delivers substandard services" [6]. More recently, XHTML has become the standard representation for both desktop and mobile web applications. With XHTML, researchers found a tension between long pages that require excessive scrolling, and sequences of short pages resulting in several high latency downloads [15].

Because there is a negligible existing application base in the rural developing world, CAM is not constrained by the limitations of current platforms or protocols. Instead of using menus, CAM applications are navigated with barcodes and numbers. Rather than optimize spatial layout of pages, CAM displays prompts in sequence to reduce screen requirements and user decisions. Instead of being written declaratively, CAM applications are scripted, generating these sequences of actions. To support offline use, the entire application is downloaded at once, rather than page-by-page. CAM also takes full advantage of the mobile phone's audio and image input and output capabilities.

SMS-based services are very popular, particularly in the developing world. CAM provides an interactive, multi-media client on top of an underlying SMS transport layer. In a different sense, CAM is similar to Interactive Voice Response (IVR) systems. In an IVR system, an online system asks the user a number of questions that he responds to by selecting a numeric option or providing a voice response. However, while IVR is implemented using a active voice connection, CAM applications work offline. CAM also takes full advantage of the mobile phone's screen and other UI features. Researchers have found the combination of IVR with a visual display to be beneficial for usability [30, 41].

8.2 Paper UIs

Researchers have commented on the importance of paper in workplace settings [36, 20], and developed new technologies linking paper and digital media [13, 11, 21, 17, 10]. None of these systems have used a mobile phone as the primary interaction device, and none provide a generalized mobile client-server development platform.

The Mobile Server Toolkit [39] is a visual tag system allowing mobile devices to access site-specific information services. The MST transmits individual UI events to an application running on a nearby bluetooth-connected server. In contrast, CAM handles all user interaction on the phone itself - messages to the server contain data elements only. CAM also works offline, sending messages asynchronously using standard protocols. These features make it more suitable for accessing remote, disconnected services.

The Cooltown project [16] used RFID, barcodes and and other sensing technologies to retrieve parameterized HTML documents. The visual code widget library developed by Rohs [31] included a declarative UI specification that links visual codes with a rich set of data entry widgets. McCune et al. propose using barcodes for exchanging trusted public keys between mobile devices [22]. Belotti et al. have developed a platform for rapid prototyping of multi-channel, multi-modal, context-aware, paper-based applications. Their system requires the additional expense of a digital pen, and does not support offline, scripted interaction [1].

8.3 Store-and-Forward Networks

Several researchers have proposed store-and-forward networks based on transport of physical media as a high-bandwidth, high-latency communications alternative for rural areas in the developing world [27, 40, 4]. Most of this work has focused on the networking requirements. Our work is orthogonal to these efforts. We propose an end-to-end mobile information service architecture that can build upon the lower-level protocols pioneered by these projects.

9. CONCLUSION

In this paper we have presented CAM: a mobile information service architecture for the rural developing world. Not being constrained by an installed application base or pre-existing assumptions about the delivery of web-based services, we have started from first principles to develop an architecture that is uniquely suited to rural device, user and infrastructure constraints.

We are currently designing, developing and deploying a number of applications using CAM. CAM has been driven from the beginning by the demands of real applications [24], and we expect the evolution of the system to come about by the same means. For that and other reasons we plan to release the CAMBrowser application under an open source license, so that the system can benefit from the contributions of other researchers and developers around the world.

10. REFERENCES

[1] R. Belotti, C. Decurtins, M. C. Norrie, B. Signer, and L. Vukelja. Experimental platform for mobile information systems. In *Proc. MobiCom 2005*, 258–269, ACM Press.

[2] S. Björk, L. E. Holmquist, J. Redström, I. Bretan, R. Danielsson, J. Karlgren, and K. Franzén. WEST: a web browser for small terminals. In *Proc. UIST 1999*, 187–196, ACM Press.

[3] E. H. Blake. A field computer for animal trackers. In *Ext. Abstracts CHI 2002*, 532–533, ACM Press.

[4] E. Brewer, M. Demmer, B. Du, M. Ho, M. Kam, S. Nedevschi, J. Pal, R. Patra, S. Surana, and K. Fall. The case for technology in developing regions. *Computer*, 38(6):25–38, 2005.

[5] S. Brewster. Overcoming the lack of screen space on mobile computers. *Personal Ubiquitous Comput.*, 6(3):188–205, 2002.

[6] G. Buchanan, S. Farrant, M. Jones, H. Thimbleby, G. Marsden, and M. Pazzani. Improving mobile Internet usability. In *Proc. WWW 2001*, 673–680, ACM Press.

[7] O. Buyukkokten, H. Garcia-Molina, A. Paepcke, and T. Winograd. Power browser: efficient web browsing for PDAs. In *Proc. CHI 2000*, 430–437, ACM Press.

[8] A. Fox, I. Goldberg, S. D. Gribble, D. C. Lee, and E. Brewer. Experience with Top Gun Wingman: A proxy-based graphical web browser for the 3COM Palm Pilot. In *Proc. Middleware 1998*, September 1998.

[9] Grameen Phone home page. http://www.grameenphone.com, March 2005.

[10] F. Guimbretière. Paper augmented digital documents. In *Proc. UIST 2003*, 51–60, ACM Press.

[11] J. M. Heiner, S. E. Hudson, and K. Tanaka. Linking and messaging from real paper in the paper PDA. In *Proc. UIST 1999*, 179–186, ACM Press.

[12] Honey Bee Network home page. http://www.honeybee.org, November 2005.

[13] W. Johnson, H. Jellinek, J. Leigh Klotz, R. Rao, and S. K. Card. Bridging the paper and electronic worlds: the paper user interface. In *Proc. CHI 1993:*, 507–512, ACM Press.

[14] A. Joshi, S. Weerawarana, and E. N. Houstis. On disconnected browsing of distributed information. In *Proc. RIDE 1997*, page 101, IEEE Computer Society.

[15] A. Kaikkonen and V. Roto. Navigating in a mobile XHTML application. In *Proc. CHI 2003*, 329–336, ACM Press.

[16] T. Kindberg. Implementing physical hyperlinks using ubiquitous identifier resolution. In *Proc. WWW 2002*, 191–199, ACM Press.

[17] 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 *Proc. CHI 2003*, 89–96, ACM Press.

[18] H. Lam and P. Baudisch. Summary thumbnails: readable overviews for small screen web browsers. In *Proc. CHI 2005*, 681–690, ACM Press.

[19] M. Lamming, M. Eldridge, M. Flynn, C. Jones, and D. Pendlebury. Satchel: providing access to any document, any time, anywhere. *ACM Trans. Comput.-Hum. Interact.*, 7(3):322–352, 2000.

[20] 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.

[21] W. E. Mackay, G. Pothier, C. Letondal, K. Boegh, and H. E. Sorensen. The missing link: augmenting biology laboratory notebooks. In *Proc. UIST 2002*, 41–50, ACM Press.

[22] J. M. McCune, A. Perrig, and M. K. Reiter. Seeing-is-believing: Using camera phones for human-verifiable authentication. In *SP 2005: Proceedings of the 2005 IEEE Symposium on Security and Privacy*, 110–124, Washington, DC, USA, 2005. IEEE Computer Society.

[23] T. S. Parikh. Using mobile phones for secure, distributed document processing in the developing world. *IEEE Perv. Comput. Mag.*, 4(2):74–81, April 2005.

[24] T. S. Parikh, K. Ghosh, A. Chavan, P. Syal, and S. Arora. Design studies for a financial management system for micro-credit groups in rural India. In *Proc. CUU 2003*, 15–22, ACM Press.

[25] T. S. Parikh, P. Javid, S. Kumar, K. Ghosh, and K. Toyama. Mobile phones and paper documents: Evaluating a new approach for capturing microfinance data in rural India. In *Proc. CHI 2006*, ACM Press.

[26] J. Pascoe, N. Ryan, and D. Morse. Using while moving: HCI issues in fieldwork environments. *ACM Trans. Comput.-Hum. Interact.*, 7(3):417–437, 2000.

[27] A. S. Pentland, R. Fletcher, and A. Hasson. Daknet: Rethinking connectivity in developing nations. *Computer*, 37(1):78–83, 2004.

[28] Wikipedia - population density, Nov 2005. http://en.wikipedia.org/wiki/List_of_countries_by_population_density.

[29] The real digital divide. *The Economist*, Mar. 2005.

[30] T. L. Roberts and G. Engelbeck. The effects of device technology on the usability of advanced telephone functions. In *Proc. CHI 1989*, 331–337, ACM Press.

[31] M. Rohs. Visual code widgets for marker-based interaction. In *Proc. IWSAWC 2005*, Columbus, Ohio, USA, June 2005.

[32] 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*, 265–271. Austrian Computing Society (OCG), Vienna, 2004.

[33] S. Rutherford. Money talks: Conversations with poor households in Bangladesh about managing money. *Journal of Microfinance*, 5(2), Winter 2003.

[34] B. N. Schilit, J. Trevor, D. M. Hilbert, and T. K. Koh. Web interaction using very small Internet devices. *Computer*, 35(10):37–45, 2002.

[35] A. Schmidt, H. Schröder, and O. Frick. WAP: designing for small user interfaces. In *Proc. CHI 2000*, 187–188, ACM Press.

[36] A. J. Sellen and R. H. Harper. *The Myth of the Paperless Office*. MIT Press, Cambridge, USA, 2003.

[37] Series 60 platform, Nov 2005. http://www.s60.com/.

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

[39] E. Toye, R. Sharp, A. Madhavapeddy, and D. Scott. Using smart phones to access site-specific services. *IEEE Perv. Comput. Mag.*, 4(2):60–66, April 2005.

[40] R. Y. Wang, S. Sobti, N. Garg, E. Ziskind, J. Lai, and A. Krishnamurthy. Turning the postal system into a generic digital communication mechanism. In *Proc. SIGCOMM 2004*, 159–166, ACM Press.

[41] M. Yin and S. Zhai. Dial and see: tackling the voice menu navigation problem with cross-device user experience integration. In *Proc. UIST 2005*, 187–190, ACM Press.