Appendix

Design Notes for the Salvation Army Cambridge Corps Drop-In Center Data Collection System

Ji-Jon Sit, Ben Leong
{jijon, benleong}@mit.edu

April 17, 2002

1 Background

The Salvation Army shelter/drop-in center needs a client information management system which can enable the staff to accomplish the following tasks:

- Quickly record client intake information, creating additional time during the intake for the collection of other client information to be used in homelessness research (such as reason for being homeless, work history, etc.)
- Efficiently collect aggregate and client-level service usage information for the various drop-in center services (meals, showers, laundry, clothing, medical care, etc.). Aggregate information will be helpful in fundraising efforts, whereas data on individual service usage will be used for program improvements and research on the problem of homelessness.
- Collect detailed demographic information from clients seeking to participate in Salvation Army programs. Record shelter stay information. Generate reports from data collected for purposes of analysis.

The Salvation Army Cambridge Corps currently employs a web-based program called ServicePoint for its client data collection. ServicePoint is considered "best in class" in homelessness information management and has been adopted statewide as part of a data collection effort led by the McCormack Institute of UMass Boston. Unfortunately, ServicePoint's web-based interface is very slow and labor-intensive, making certain tasks (such as collecting service usage information) utterly impractical for a large shelter such as The Salvation Army.

Unfortunately, the ServicePoint platform is inadequate for the critical tasks listed above. ServicePoint is capable of facilitating client intakes, but the labor-intensive interface (further slowed down by the wait-time to post information to the internet forms) makes collecting only the most basic intake information practical. Though ServicePoint could in theory be used to track individual service usage, this is utterly impractical, since it would require manually entering hundreds of services into the slow system each day. The Salvation Army does currently use ServicePoint to record shelter stay information, though this information must be entered manually through a cumbersome interface and can take up to an hour per day. ServicePoint is an adequate platform for collecting detailed client demographic information during lengthy needs-assessment

interviews. These interviews are only conducted for a fraction of our clients, however. Many clients that use the drop-in center and emergency shelter never participate in such an interview, which is strictly for clients interested in participating in one of our long-term stay programs.

After significant experience with ServicePoint, the staff at The Salvation Army see a clear and significant need for an improve information management system. It is envisioned that after the system is deployed, each homeless person (client) who uses any services would be issued with some form of identity (ID), perhaps some barcoded card or smart card. At registration, the client's information would be recorded by a staff member and stored in the database. Each time the client uses a service offered by the Drop-In Center or Homeless shelter, a staff member will scan the ID and record the usage. The aggregated data can use then be used to compile statistics on the various services as well as correlate consumption to the client demographics.

Why is this important? Although social welfare agencies like the Salvation Army may have strong suspicions that certain underprivileged groups really need more support and funding than they are currently getting, it is impossible to bring about any change in the status quo without the backing of cold, hard data.

The data collected from a shelter like the Salvation Army can show if, say, mental health patients actually comprise a largely overlooked homeless population. Addressing mental health may be more important than the state currently acknowledges. Or, homelessness may really be more of a problem with education – and that improving a certain stage in the educational ladder with an eye on groups-at-risk could help allieviate homelessness more than tackling homelessness by itself.

Hence ServicePoint. However, the data funneling into the ServicePoint database from across the state is crippled and useless if we cannot collect and report it accurately. Also, the present system for data transfer to ServicePoint is via a web interface, which is both labour-intensive and slow. The new system is anticipated to bring the following benefits to the Salvation Army homeless shelter:

- Allow data to be captured more efficiently and accurately than the current sign-in sheet system, which is both error prone and inefficient.
- Allow the Salvation Army to maintain its inhouse database, capturing information which ServicePoint does not capture (e.g. the number of meals served and the clients served).
- Allow for a faster, more efficient and less labor-intensive transfer of data from the Salvation Army to the ServicePoint database.

This system for the Drop-In Center will be a proof-of-concept for automated data collection. If the system works, there is a significant possibility that it will be adopted at other Salvation Army shelters nationally and internationally, and potentially at non-Salvation Army shelters as well.

2 Design Considerations

The following are some important considerations which will govern and guide the design and implementation of the new system:

- An existing application called ServicePoint is currently used for data collection purposes. The new system must be able to complement ServicePoint and integrate/share data with ServicePoint.
- The client machines are likely to be a mixture of Windows variants and possibly even be Macs, so any deployed client programs must be able to run on all these platforms without major porting/re-coding.

- The database application has to be sufficiently modular to enable new services to be added and integrated with existing data. For instance, if a client is a registered user of the Drop-In Center and later a module for the homeless daycare services is developed for the system, his biographic data should be available to the new module (i.e., for statistical/report generating purposes).
- Maintainability is a key consideration since the development team will be handing the system over to the Salvation Army or McCormick Institute for maintenance once the system is deployed. There will be a need for all programmers to adhere to a coding convention and interfaces must be clearly defined. All codes must be well-documented and commented.
- For the clothing store, it might be good to allow requests to be captured and to be able to record exactly what items are given out. For the latter, perhaps bar-coded stickers may be printed and pasted on the items so that they can be scanned as well.
- We will need to design our very own barcode encoding. In designing code, there is a need to ensure that the address space is sufficient to support any future needs.

3 Preliminary System Design

3.1 Overview

The development team will be divided into 2 groups - "interface team" and "database team". The groups will be led by Ji-Jon Sit and Ben Leong respectively. Broadly, the interface team will be in charge of everything on the client Windows PCs, while the database team will be in charge of everything at the server end.

At the client PC machines (running Windows), the interface team is expected to develop and manage the following 3 components:

- Visor (Palm OS) Software - To allow the visor to scan and collect barcodes and transfer the necessary data to the client PCs.
- Client PC (Windows) Interface Software - Simple program that runs on the client PCs which would transfer the scanned data from the visor to the database through ODBC.
- Barcode printing software - To print the bar-coded tags (on stickers).

We are likely to run Windows 2000 on the server and the server will have 2 main components:

- Microsoft SQL Server - To store all data and respond to SQL queries.
- Apache Web Server (with php support) - To run a web application which acts as the interface to the database.

Figure 1 shows an overview of the proposed system. Standard interfaces, i.e. http and ODBC were chosen to ensure that individual components could be replaced if necessary without requiring changes to other components.

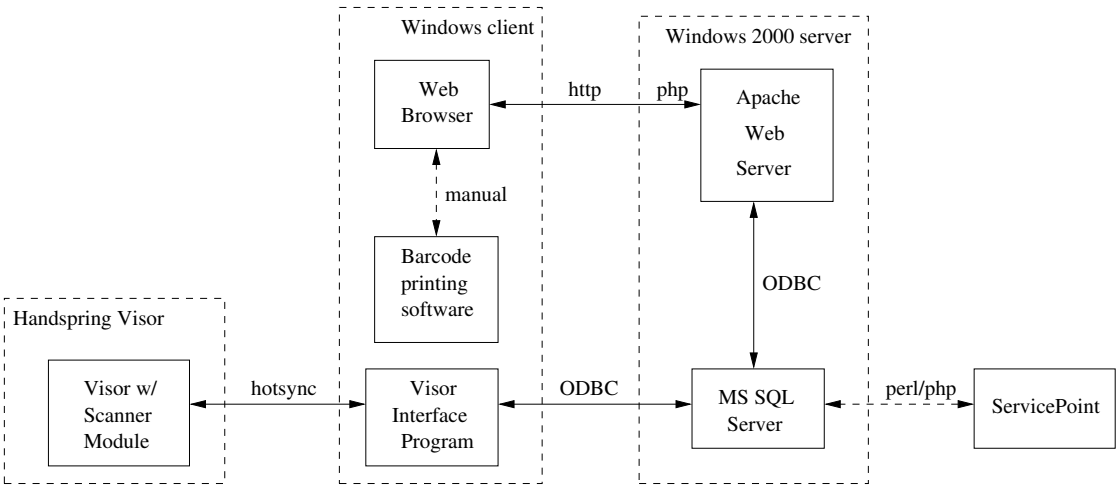

Figure 1: Overview of System Design

3.2 Client Identity Card

We considered the possibilities of using a bar-coded card, a GEMS smartcard or a RF-detectable smartcard. The Director of Operations had indicated that it would be preferable if the device to be used to scan the IDs could be portable and carried by the staff at all times. Also, the device should also be general purpose, in that it can be set to capture data for all the available services. For instance, a staff would use this device to scan IDs as food is collected by the clients; a while later, he/she could be standing outside the clothing store handing out clothes and using the same device to record the items distributed.

With these requirements and with due consideration to availability and cost, we have decided that the Handspring Visor with a bar-code scanner module would be selected for development of the prototype system. At the end of each day, the Visor would be synced with a PC and the information aggregated into the main database server. As the interface used for the database is a fixed standard (ODBC), the data-collection component may be replaced with another technology, i.e. RF-detectable smartcard, if such becomes available and its deployment becomes feasible.

3.3 Registration Host & Barcode printer

It is likely that registration will be done in one of the following ways:

- A set of cards would be pre-printed. After entering the client information into the computer, the staff worker could just enter the code for the preprinted card and that barcode number would be attached to the record.
- We can have preprinted sheets of stickers, and we can just take a sticker off quickly and put it on the card. This also has another advantage. The preprinted sheet could have 2 of each sticker. That way if there were 15 people standing in line at once waiting for a card, we could pass out forms for the clients to fill out while they are in line with their basic demographic information. When they get to the front of the line, we put one sticker on their card and one matching sticker on their form (which we then keep). Then later on, during a quieter time, we could manually enter that information into the computer and scan the sticker on the form to attach the appropriate barcode to the record.

It would probably be good to integrate the 2 steps without requiring a human to type in the numbers all over again and reduce errors. Also, it is clear that the web interface may not be flexible enough for us to deploy the most user-friendly interface. However, upon careful deliberation, we decided that the following two factors make the web interface the preferred choice for the prototype system:

- It is platform independent. Regardless of the client machine type (whether Windows or Mac), a web browser will definitely be available.
- Ease of deployment. Essentially, in case of upgrading, there is little overhead in deployment because only the server has to be modified. If we ran scripts on the clients machines, some way has to be devised to distribute application updates which will complicate our lives at this point.

In any case, it is clear to us that the most important component of the project is not the database interface, but the data collection and database integration system. When the initial system is deployed, we can possibly replace the web interface with some client-side scripting program (perhaps in Ruby) which can communicate with the Microsoft SQL Server directly through ODBC.

3.4 Scanner

This is the most important component of the project. We will require software on the Handspring Visor to allow us to scan and store the data scanned from the barcodes on the ID tags. It is not likely that the available software for the Palm OS will be able to support the user interface we need so it is likely that we will have to write our own application to do the work.

The interface on the Visor envisioned is one where the user should be able to click on a little button which indicates the service type, say "lunch" or "laundry". Then, perhaps the user will then hold the scanner over a barcode and then press one of the buttons to scan. The application will then log down the user ID, service provided and date/time.

In the case of clothing supplies, there are 2 possible approaches. In the first approach, the user selects "clothing" and scans the ID. Then, the user manually enters all the items given to the client using the interface. In the second (and we believe preferable) approach, all items are also barcoded, so the user scans the ID and then proceeds to scan the barcodes on the distributed items and the application will record the user ID, item distributed and date/time automatically. The application will also have to understand the encodings for *clients* and *items* so that it can do the right thing.

3.5 Synchronization Program

A simple windows application would have to be written to synchronize with the Visor and to transfer the collected data directly to the Microsoft SQL Server directly through ODBC. We have not decided on developmental platform for this application. The choice would depend largely on the comfort level of the programmer assigned to this component, perhaps Visual C++ or Visual Basic?

3.6 Server

We intend to run Windows 2000-based server and install both Apache and the Microsoft SQL Server on the machine. All interfaces with the SQL Server will be through ODBC and since Apache has native php support, the database application will be developed in php.

Two servers will be set up for this project. The production server will be set up on the Salvation Army Cambridge Corps network, while the developmental server is likely to be set up at LCS, since most of the developers are likely to be based in LCS. When the project is completed, the development server will be moved onto the Salvation Army network as set up as a backup server for the production server.

3.7 Interface with ServicePoint

ServicePoint is a hugh database program. Although the Salvation Army only uses a subset of its functions, it would be prudent for us to import the entire schema of the ServicePoint database to allow for easier data interface between the two systems. We will need to discuss with Bowman (the software company that developed ServicePoint) how they can share their database schema with us and how data can be uploaded to ServicePoint database. We do not expect them to grant us direct Internet access to their database and hence it is likely that information would be exchanged in the form of an interface file. Once the format is decided, it is likely that we will write a script (perhaps in perl) to extract the information from the database to generate the required file. It is possible to use php as well and integrate the file generation with the Registration program.

In addition to the ServicePoint database schema, we will also have to develop an in-house schema to support the capture of data which we will require locally and is not captured in ServicePoint. It is anticipated that this will require a substantial amount of effort as care must be taken to ensure that the schema designed is efficient and yet modular enough to allow new services to be integrated into the system without code modifications, if possible.

4 Anticipated Requirements/Budget

The following is the breakdown of the expected budget (approximately $4,000) for the major items for the project:

Item	**Cost per unit**	**Quantity Required**	**Total**
Development Server (1.5 GHz Intel Pentium 4, 40 Gig HD, 512 MB RAM 10/100 Ethernet, Windows 2000, 40xCDROM)	$1,000.00	1	$1,000.00
Production Server (1.5 GHz Intel Pentium 4, 40 Gig HD, 256 MB RAM 10/100 Ethernet, Windows 2000, 40xCDROM)	$900.00	1	$900.00
Handspring Visor (Reconditioned)	$100.00	3	$300.00
CSM 150 Barcode Scanner	$150.00	3	$450.00
SPT 1500 Pocketable Computer (4MB)	$400.00	1	$400.00
Barcode Printer	$500.00	1	$500.00
CodeWarrior for Palm OS	$400.00	1	$400.00
Microsoft SQL Server 2000 (Production Version)	$1200.00*	1	$1200.00*
Microsoft SQL Server 2000 (Development Version)	$500.00*	1	$500.00*

*Not included in budget.

Since we do not not have a budget, we intend to approach the respective companies to see if they would be willing to donate some equipment for a good cause or give us a discount on their products. With regard to the Microsoft SQL Server 2000, we plan to seek a donation from Microsoft. If we are unable to secure a donation, we will run instead run a postgres server with ODBC drivers on Linux.

5 Projected Project Schedule

The current projected schedule for the project is as follows:

Phase	Date	Activity
I	1 Mar - 30 Apr	(a) Apply for grants, write to companies for sponsorship (b) Preliminary system design and mustering of manpower
II	1 Apr - 14 May	(a) Acquisition and familiarization of available hardware/software (b) Work out database schema (c) Set up development and deployment servers (d) Refine overall design & define interfaces
III	1 Jun - 15 Jul	System Development
IV	16 Jul - 15 Aug	System trials
V	15 Aug onwards	Prototype System fully deployed and functional (hand over to Salvation Army)

6 Volunteers

The current list of MIT graduate student volunteers who are involved in the project is as follows:

S/N	Name	Department	Email
1	Indraneel Chakraborty	Computer Science	indranil@mit.edu
2	Ben Leong (leader: database team)	Computer Science	benleong@mit.edu
3	Cynthia Lo	Chemical Engineering	clo@mit.edu
4	Lik Mui	Computer Science	lmui@mit.edu
5	Steven Richman	Computer Science	richman@mit.edu
6	Archit Shah	Computer Science	ashah@mit.edu
7	Ji-Jon Sit (leader: interface team)	Electrical Engineering	jijon@mit.edu