Real-Time Mobile Data Access to Enterprise Networks

Florida Gulf Coast University
Real-Time Mobile Data Access to Enterprise Networks

Team Members
Anders Petersen, CS major: ahpeters@eagle.fgcu.edu
Dustin Felton, CS major: dgelton@eagle.fgcu.edu
Sean Kerwin, CS major: spkerwin@eagle.fgcu.edu
Chad Singleton, CS major: cjsingle@eagle.fgcu.edu

Team Mentor
Dr. Janusz Zalewski: zalewski@fgcu.edu

Mentor’s Assistant
Ray Rhode: rarhode@eagle.fgcu.edu
Abstract

The goal of this project is to provide a comprehensive set of solutions to allow the extension of a network’s resources beyond the physical confines of the network; effectively, to let a corporation’s infrastructure break through the walls of the building and extend all over the world [1-2]. This goal is accomplished by a number of open-ended sub-items, each highly customizable and extensible, which provide the necessary means to extend the reach of a variety of business activities.

The precise implementation of these technologies is inherently tied to the nature of an organization; clearly a corporation with no manufacturing facilities has no need of linkage to a factory. As a result this team has elected to implement each component of this broad system with a single representative organization in mind, rather than attempt to concoct a hypothetical organization with full need of all of these technologies.

The first primary component of the project is a spatially cognizant portable service location and activation system, hosted on a cellular phone. The implementation presented here was designed in cooperation with GuestClick, Inc., and is a generalized service reservation broker: a hypothetical user in need of a hotel reservation, for instance, could automatically locate all proximate hotels and reserve a room, and then make dining reservations and reserve a rental automobile using the same system. The project component has been termed the Web-Based Transaction client, or WBT.

The second primary component of the project is a system allowing for the real-time monitoring of multiple data sources using a standard Portable Digital Assistant (PDA). The demonstration implementation integrates with technologies by USA Technologies and allows for the remote monitoring of inventories in multiple beverage vending machines, allowing a delivery person to optimally schedule/prioritize restocking trips.

The project also incorporates various other technology shims that allow for the integration of other existing devices and technologies into a unified corporate architecture. Specifically, information is provided concerning use of the LabView virtual instrumentation software, as well as a generic CORBA-to-SQL bridge. It is intended that these components be used as glue to bind new facilities to a network or to properly connect the two primary components to existing facilities. They can therefore be viewed as a generalized expansion point for the architecture.
1. System Overview

1.1 What the System is Meant to Accomplish?

To achieve the project’s objective, remote data access to enterprise networks, we provide the capability of accessing enterprise-wide systems from a remote location or a vehicle, both for customers and employees. The overall view of the system is as follows:

- when the focus is on the customer access to obtain services, we developed a cell phone location-aware application for business transactions (using hotel reservation as an example, in cooperation with GuestClick Inc.)
- when focus is on the employee access to conduct business on the road, we developed a wireless PDA application for remote vending access to machines (using a coke machine as an example, in cooperation with USA Technologies)
- when focus is on expanding the business to integrate new functions into the system, we developed an application to incorporate remote robot control (in cooperation with Robotic Workspace Technologies Inc.)
- when focus is on increasing business integration via a multi-purpose enterprise-wide network, we developed a CORBA based framework for remote data access.

1.2 Performance Requirements (for Cell Phone and PDA Applications)

Wireless Business Transaction Client/Server. The Wireless Business Transaction (WBT) component of the project shall be required to handle at least 100 simultaneous users with each user experiencing no more than 5 seconds of server-related latency. It should be noted that total latency will likely be far greater than this as a result of transport latency and the speed of the client applet's host.

PDA Vending Machine Client/Server. The PDA Vending Machine client is designed to send/request queries to the database periodically throughout the day. The PDA will be wireless using Ethernet, so a request shall return results with no more than 5 seconds of server-related latency. The client shall be able to store at least 15 vending machine record sets in its local memory. The server shall be able to support up 1,900 simultaneous users.

1.3 Summary of the Design Methodology

Our design methodology is immersed in a business model of product development, of which the technical development itself is just a part [7-8]. The process involves the following five stages: Market Analysis, Technology Selection, Education and Training, Development, Product Marketing and Sales. The Market Analysis stage has involved potential local customers, who provided data supporting the need of developing the product:

- Coca Cola, USA Technologies and Harvey Software for vending machine [3-6]
- GuestClick Inc., for interactive business transaction services [11-12]
- RWT Technologies for remote robot access [9-10].

Technology Selection is described in the Tradeoffs and Optimization section, and Education and Training was done during the weekly meetings.
We based the Software Development cycle on a traditional, simplified waterfall model, in a spiral version, with emphasis on the Testing phase. This model is very effective for small projects with very limited duration, such as ours. Project schedule and team responsibilities involved 4 typical Milestones (Requirements Spec, Design Description, Implementation and Testing) spread over the 4-month project duration (one month each), with one team member responsible for a single milestone. Two members were focusing on cell phone application development and the other two on the PDA application.

1.4 Project Novelty and Originality

This project is unique on a number of levels. Taking a broad view, this project is unique in that instead of focusing on the creation of a single product or service it attempts to incorporate features useful to an entire organization. This decision is easy to justify: by offering a single comprehensive solution for the expansion of networked infrastructure, this project has the potential to benefit companies of all sizes, both by speeding expansion and integration and by lowering costs. Further, the subsidiary products, the CORBA and LabView components of the project, provide means of creating linkages between companies or between units of a single company. Though software exists for inter-corporate linkage already, most are largely proprietary and difficult to integrate with existing systems. The LabView and CORBA components of this project can be bound to nearly any existing system without substantial reengineering effort. At the low level, the individual components implemented to demonstrate the project are original as follows:

*Location in WBT Client.* The cellular phone client is unique in that it uses physical location as part of the electronic service location process. Although facilities such as hotel reservations or car rentals have been available from small-device clients for quite some time, the user interaction dynamics of such clients have always been complicated by the need to manually select a provider. This marks a significant step in usability.

*Expandability/Flexibility of WBT Server.* The WBT server is designed such that results are formatted using an Extensible Stylesheet Language document; essentially the formatted output is the product of an externally-defined transformation. This allows the output of the server to be modified for different applications without any actual modification to source code. Though XSL is obviously a preexisting technology, research has not revealed any other instance of its application to data output of server side scripts.

*PDA Client – Potential Shipping Application.* One potential application investigated for the PDA component was package tracking. If put to such a use, the application would be a novel solution compared to existing systems as used by UPS or FedEx, for instance. Those system track delivery and inventory only when the driver returns to a distribution center; this system could update tracking information immediately as a package is delivered.
2. Implementation and Engineering Considerations

2.1 Detailed Specification of the System

2.1.1 Requirements for Wireless Business Transactions via Cell Phones

The objective of this part of the project was to develop and test an application allowing business transaction processing from a cell phone, to make hotel reservations as an example. The general concept is presented in Figure 1. A cell phone serves as a client accessing information via the Internet from a remote server that in turn gets the service list and necessary data from an information provider.

A wireless business transaction is understood as a request, made from a cell phone, for a particular service to an information provider, and a response from information provider involving possible interaction leading, under normal circumstances, to completion of an order. A complete transaction is composed of three phases, illustrated in Figure 2, all initiated from a cell phone, as follows:

(a) Initiation Phase – a general request for a service made from a cell phone to the intermediary server. The intermediary server asks the information providers for the necessary information, verifies and modifies the data to suit the client and transmits the data to the client.

(b) Selection Phase – a specific request for detailed information, based on the initial data, made from a cell phone to the intermediary server; this request results in a display with specific data, including service parameters, pricing, order form, etc., delivered by an information provider via the intermediary server.

(c) Completion Phase – a purchase order made from a cell phone, resulting or not in confirmation of an order.
Additionally, there is a setup phase that is executed the first time the client runs. This consists of a request is sent from the client to the server asking for a list of transaction types the server has available. If possible, this list is then stored locally on the client so that it can be reused. For client implementations where this is not possible (i.e. WAP/WML), the setup phase will have to be made every time the client runs. This list can also contain information about additional information the server requires to complete the transaction. Although this system is designed to be able to handle multiple transaction types, it can easily be configured to only allow one type of transactions if that is desired.

Specific requirement for the client and server parts are omitted due to space limits, but can be access via project website.
2.1.2 Requirements for Real-Time Vending Machine Updates via PDA

This part of the project entitled creating a software package that is able to give vending machine service providers an up-to-date reading of the status of their machines’ information wirelessly from their PDA and let them upload, in real time, it to a database [3-6]. The system architecture consists of three layers, which are the information providers, intermediary server, and client. The general concept is presented in Figure 3.

The information providers level will be responsible for updating the intermediate server with the information gathered from the vending machines. This layer represents the organization of information gathered from a company’s Data Acquisition Devices, which in our case is USA Technology’s ePort. It is the responsibility of the information providers to be able to properly gather the information and output it into a format Comma-Separated Value (CSV), is parsable by the intermediary server.

![Figure 3. Overview of PDA Application System Architecture.](image)

The next layer is the intermediary server, which translates the data from the information providers into an Internet accessible database. It is responsible for handling queries from the client of specific vending machine information such as: inventory, temperature, sales, etc. Having the information stored at this layer instead of at the client level reduces the amount of memory overhead for the client, which is usually limited to about 128 Mbytes.

The client layer provides the users with a software application as an interface to the intermediary server’s vending machine database. It will be used to directly by users to make decisions in the field such as which machines need repair, or restocking. The client software will allow the user to specify preferences such as the order in which the vending machines are viewed and whether or not to show only machines that need restocking.
2.2 System Design

2.2.1 Cell Phone Application: Wireless Business Transaction Client and Server

Client Architecture. Client is a finite state machine which maintains three states: previous, current, and next. Each state is representative of a single user interface view. Having three states stored makes reuse of user interface elements possible given the limitations of the API. A state devoted to receiving a ZIP code from the user can be used either during a service query or during preference configuration, but will be the same in either case; the only difference will be the next state.

Likewise, the previous state allows for the proper handling of validation. During initial configuration, the user must enter his/her user name, whereas later while changing preferences the user may cancel the operation. In the first case, the previous state would be an exit state; if the user fails to complete first-time initialization, the program exits. In the second case, the previous state would be the preferences screen.

Client uses the RecordStore class to maintain necessary information. Rather than concern itself with the RecordFilter and RecordComparator, the client will simply decide in advance which record ID is assigned to what information.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Username (Unicode string)</td>
</tr>
<tr>
<td>2</td>
<td>Location Mode (integer)</td>
</tr>
<tr>
<td>3</td>
<td>Default Location (Unicode string; format specified by {2})</td>
</tr>
<tr>
<td>4</td>
<td>Last Service Type List Update (Date)</td>
</tr>
<tr>
<td>5…N</td>
<td>Service Type List (encoded as Unicode string)</td>
</tr>
</tbody>
</table>

Because RecordStore information is application-specific, there is no way a future version of the application could encounter preferences stored by a previous version. Though this is unfortunate from a usability perspective, it does make it unnecessary to future-proof the preference storage.

![Figure 4. Cell Phone Application Server State Diagram.](image)
Server. The Wireless Business Transaction Server consists of two main parts, the application server and the user server. The application server handles all the communication with the clients while the user server handles account management and would basically be an integrated part of the website for those that choose to implement the WBT system. The application server is a finite state machine with 4 states (Fig. 4).

Information necessary to identify which state the client request will be transmitted every time a request is made. By doing that, we avoid having to use sessions and it greatly simplifies the design of the server. Output from the application server is delivered across a HTTP connection established by the client. The client can choose whether the output should be XML or CSV. CSV, though, is the default output mode as it offers the most bandwidth efficient method for clients on limited mobile devices. Not only does XML use more bandwidth (and hence increase user costs), but it is also considerably more difficult to parse with the limited resources available on for example a cell phone.

Implementation. Due to space limitations do not show part of the pseudocode for the application server. It is available from the project website. Software has been implemented on the Siemens S65 cell phone with the following parameters: wireless technology: GSM/GPRS, network frequencies 900, 1800, 1900 MHz, memory 10 MB, software CLDC 1.1, MIDP 2.0 (also supports: JTWI 1.0, M3DAPI 1.0, MMAPI 1.1, LAPI 1.0, JABWT 1.0, WMA 1.1), additional connectivity: BlueTooth, Infrared; screen: 132x176/16 bits. Sample screenshots from the cell emulator are shown below.
2.2.2 PDA Application: Wireless Data Access to Vending Machines and Databases

The PDA Wireless Access Design consists of three main parts (Fig. 5), the client application, the server-side middleware application, and the database. The client application is responsible for display information gathered from the database via the middleware application.

Figure 5. System Architecture of the PDA Application.

Client - Presentation Layer. The Client application is a Java application written for a Java 2 Micro-Edition (J2ME) compliant Virtual Machine (VM). The particular Virtual Machine currently running our client application is a shareware version of NSICom’s CrEme version 4.0. The application, when run by the Java interpreter on a Linux operating system, functions correctly. Sample pseudocode for the presentation is shown in below.

```java
NewFrame {
    new SCHATcomm()
    establishServerConnection
    retrieve usernamesList
    show login frame
    send username/password
    if (driverMachineList exists)
        showDriverMachineList()
    else
        list ServiceableMachines()
    if (userAction == button1)
        showProductReq
    else if (userAction == button2)
        markMachineAsServiceable
    else if (userAction == button3)
        showMachineDetails
}
```
Server - Business Rules Layer. The business logic/database connectivity layer is responsible for the following:

- Retrieving and formatting data from information provider (USA Technologies)
- Use business logic to interpret information provider’s data
- Feed the intermediary server’s database the interpreted data
- Sending and receiving request between the client and database

This layer is also needed for database interaction between the client and the database. Although a Java application can directly access a remote database, it requires an additional driver to be installed. In the case of the PDA, this would require installing the
database driver on each device. To avoid extra software maintenance we connect the client to a server-side application that uses a database driver stored on the server.

This layer consists of two main parts (Fig. 6), the process that deals with feeding data from the information provider to the intermediary server database, and the process that deals with sending and receiving requests from the client to the database and vice versa.

**Database.** The Database class is a Java Class (Fig. 7), which contains all the methods for querying and accessing the database as required by the Server application. In order to support Database queries from Java, we had to build a database to contain all of the vending machines’ information. The DBMS we decided to use was MySQL. It was determined that the database would handle employee, login, and route information in addition to the vending machine information. The table definitions were created using MSAccess, and then converted to a MySQL format using Intelligent Converter’s Access to MySQL program.

USA Technologies gave us access to a demo account with 3 months worth of sample data from more than 10 vending machines. Data from the demo account was limited to custom report which generate csv (comma-separated) files. Had we had access to a live account we would access to more detailed information in the form of a DEX file directly from the ePort. Having gathered sample data from the USA Technologies database in the form of csv files we were able to get a general idea of what types of information we could include in our client software, and what additional data manipulation would have to be done on the intermediary server. Also, because the demo account uses machines that are no longer transmitting live data, we need to be able to simulate how our program would process new files that would be generated daily from USA Technology’s server.

[Figure 7. Class for Database Structure]
**Implementation.** The client was implemented on a Dell PDA AXIM x50v, with RAM of 192MB (152MB usable), Intel xScale running @ 624Mhz (ARM architecture) processor, Windows 2003 Mobile Edition (CE 4.21.1008) operating system, Java Virtual Machine NSiCom (www.nsicom.com) CrEme v4.0 (supports CORBA, J2ME CDC 1.0 spec, based on SDK 1.3.1). Sample screenshots from the application running are shown below.

2.3 Tradeoffs Involved: Optimization of Design and Implementation

2.3.1 Cellular Phone WBT Server

*Server Design.* The decision was made to implement the server as a PHP-based web application instead of developing it as a proprietary architecture mostly because of the reliability that comes from using proven technologies such as PHP and Apache. This decision reflected the belief that project goals could be adequately accomplished using these existing tools, and that use of widespread and well-tested technologies would provide the WBT server a solid basis for reliable real-time performance. This decision complicates the design of the server, however, in that it is difficult for a service accessed via multiple CGI requests to maintain state.

*Networking Protocol.* Data is transmitted from the server to client via HTTP as a series of command separated values (CSV). Early prototypes were constructed using XML transport, but CSV was selected over XML for the final design for two reasons. Firstly, CSV results in a far smaller data size, which is highly relevant in an environment where many users will be paying by the byte. Secondly, it was discovered during development that on some clients XML parsing was unfeasible due to processor speed and memory restraints. This decision has no significant drawbacks outside of the potential marketing ramifications of abandoning the popular XML technology.
**User Interface.** Given the choice of HTTP as the transport mechanism, there were two paths available for the client interface. The easiest choice, a webpage in WML, was rejected very early in the design phase due to the limited usability and compatibility it entailed. Design thus passed along the second path, a custom Java interface. As AWT is unavailable on most cellular phone clients due to their support of only the J2ME Midlet profile, the only option was the lcdui library. Though this limits the usability of the interface it is the only feasible option short of complete implementation of a custom user interface in graphical mode.

**Location Determination.** Several different methods of determining the location of the user were researched. From the user perspective, the easiest option is for his or her location to be determined automatically. This is possible on phones supporting the JSR-179 Location API, a recent addition to the J2ME Midlet specification. Unfortunately research indicates that no cellular service providers in this country support the Location API on their phones. The decision was therefore made to allow users to alternatively specify ZIP or area codes during a query. Though this concession is a clear loss to user-friendliness, it allows for functionality on current technology while supporting improved operation in the future.

### 2.3.2 PDA Component

**Platform.** It was decided early in the development of this component to use Java as the primary platform. This allows for easy portability between different PDA platforms (Windows Mobile and PalmOS). Though the portability is a major benefit to the project, it carries the penalty of requiring the installation of a Java Virtual Machine (JVM).

**User Interface.** The interface for the client component of the PDA component was constructed entirely using the Java Advanced Windowing Toolkit (AWT). Though the newer Swing library provides a superior user experience and is in many respects an easier API with which to develop, our survey of available small-footprint JVMs demonstrated that Swing support can be provided on very few devices and is only partially implemented on those.

**Networking Protocol.** The decision was made early on to use socket communications between the PDA client and server, as opposed to using HTTP (as the cellular client/server pair do) or CORBA. The decision to reject CORBA was trivial, as the availability of the technology on PDA clients was nearly as lacking as that of Swing (above). The decision not to use HTTP was largely based on reliability, security, and ease of server implementation.

**Choice of JVM.** The NSI CrEme JVM was ultimately selected for the development platform of the PDA client. Though it is less than comprehensive in its Java support, it meets the minimum needs of the project and meets pricing requirements. The alternative of the Blackdown J2RE was discussed but ultimately rejected, as it would required
installing a Linux implementation on the test PDAs, and therefore would have represented a relatively unusual test case.

2.4 Description of Tools Developed

*Educational Tools.* During the course of the project it frequently became necessary for team members to research a particular technology and summarize it to the group. These presentations were prepared with the intention that they be generalized and not project-specific, thus they have potential pedagogical or education value. The presentations and associated sample code has been made publicly accessible http://csidc.fgcu.edu/presentations.php.

*TestMonkey Automated Testing Tool.* Performance of the intermediate WBT server was tested using a customer Mac OS X based tool called TestMonkey. TestMonkey accepts settings describing a command line executable and the parameters to be passed thereto, and executes and times a user-specified number of instances of that process simultaneously. Though TestMonkey was used in a relatively simple fashion during this project, it could be applied to a broad array of testing situations where a Mac OS X host is available.

*Eport CSV Conversion Tool.* The data provided by the Eport system is in the form of comma separated values (CSV), which must be imported into the Java server before use. The class used to implement this feature, CSVParse, has been designed for reusability. It is not dependant on any other project classes and thus can be added to any project with the need to parse CSV-formatted data with only minimal difficulty.

2.5 Verification and Testing

Because the primary components of the project (the WBT and PDA clients) are user-facing devices, responsiveness becomes a key element of the performance criteria.

*WBT Server Performance.* Performance of the WBT server was measured with the TestMonkey utility, launching several simultaneous instances of the Curl program and measuring the response time for each. To eliminate variability in networking performance, the test was run on the same machine that was hosting the server. Consequently it is reasonable to assert that the measured results test the performance of the project code and the Apache web server exclusively.

Figure 8 shows the results of a simple query returning the list of supported service types. The exponential shape of the curve is a likely a result of the unoptimized nature of the code and the fact that the Apache server has not been optimally configured. As each request requires that a new PHP interpretation context be instantiated, it appears that memory use scales with query volume, and that this particular server has only enough RAM to handle approximately 60 simultaneous requests before virtual memory paging begins to significantly impact performance.
Fig. 8. Cell Phone System Response Time for Service List Requests.

Fig. 9. Cell Phone System Response Time for Provider Service List Requests.
Figure 9 demonstrates a more complex query, requesting a list of providers available for a particular service. The curvature of this graph is far more linear than that of Figure 8, likely because the request to the service provider server is such a high-latency operation it dominates the result time and approximates a linear result.

**PDA Server Performance.** Because the PDA project is primarily concerned with the transfer of information from a database to the client, and performs few if any operations upon that data, performance is best gauged relative to database access time. The graphs below (Fig. 10-12) show the time needed to complete a sample transaction, as measured by the client software, compared to the time needed to complete the database access, as measure by the server software.

**Database Stress Test.** This graph (Fig. 10) examines the processing time of MySQL on two different platforms with a large dataset. Because MySQL can run on a Windows on Linux machine, it seemed appropriate to demonstrate it on both platforms. A large dataset was used to observe how each Operating System would handle the socket requests and processing time of the system. Although, the Windows server had a 3.0 GHz processor compared to the Linux server’s 1.0 GHz, there is very little evidence to concur that either operating system or processor speed is significantly faster.

![Avg CSV to MySQL Time Per Record](image)

**Figure 10.** Platform Dependency for MySQL Database.

**Business Rules Processing vs. Database Update Times.** This graph (Fig. 11) focuses on a small set of 100 records to compare data conversion to data transport between the business layer and the data layer. According to our study, the business layer counts for nearly 67% of the total time. Because of this, special attention must be paid to this layer to ensure that the performance expectations are met. Luckily, the three-tier architecture allows for distribution of this layer to multiple resources.
Client Connectivity/Performance Test. The next graph (Fig. 12) shows how long it took for the server to receive the connection request from the client app after the app was started. It’s marked "Connection time". Another bar on the same chart shows how long the program itself took to load completely after being started (marked "Load Time"). The connection graph was created to give an indication of how long, on average, one can expect for requests to be acknowledged and accepted by the server. Since all requests are handled the same way as the initial connection this average connection time reflects sending and receiving of data to and from the client app. The load time is just a measure of performance for the app on the PDA itself.
Summary

The goal of this project was to provide a comprehensive set of solutions to allow the extension of a network’s resources to help customers and businesses use their wireless resources in business transactions.

Two essential specific objectives have been achieved:

- customer access to business transactions (hotel reservations) has been provided via cell phone application, according to specifications by GuestClick Inc.
- employee access to vending machines and enterprise networks has been provided via PDA application, according to the specification by USA Technologies.

Additionally, a request for adding connectivity to an enterprise to a network for a remote application running a robot has been also provided (according to a specification by RWT Inc.), which proves the flexibility of the solution. Due to the lack of space, no details are included in this report, but can be obtained from the project website.

The project also incorporates various other technology shims that allow for the integration of other existing devices and technologies into a unified corporate architecture. Specifically, information is provided concerning use of the LabView virtual instrumentation software, as well as a generic CORBA-to-SQL bridge. It is intended that these components be used as glue to bind new facilities to a network or to properly connect the two primary components to existing facilities. They can therefore be viewed as a generalized expansion point for the architecture.

The parts of the project described in this report are nearly completed and are ready for demonstration. Future work will include adding new applications for cell phones and PDA’s and further expansion towards connectivity through such technologies as LabVIEW and CORBA.
References


[8] Presentation by Ryan Stephens, FindWhat.com, Fort Myers, Florida, April 1, 2005

[9] Presentation by Walt Weisel, RWT, Fort Myers, Florida, April 15, 2005


[12] Presentation by Henddehr Pedroza, Motorola, Plantation Beach, Florida, March 11, 2005