Reflections on a Systems Integration Project for Internet Banking

by
Bernard S. Hirsch

Hewlett-Packard Company
930 East Campbell Road
Richardson, Texas 75081 USA
(972) 699-4197

bernie@fssc.hp.com
http://home.sprynet.com/sprynet/bernie06/intbank.htm [External Version]
http://fssc.hp.com/PAPERS/internet_bank.html [HP Internetal Use Only Version]
August 1997

Preliminary Draft 0.95

Abstract. This paper will benefit IS managers, bank business managers, systems integrators, consultants, and other persons interested in helping their corporations interact with their customers over the Internet. Instead of theorizing over a hypothetical scenario with marketspeak, as is commonly done, a real business project is described. Specifically, a large U.S. retail bank is enabled to start delivering banking services over the Internet. Both the process and the technology for creating the bank's new Internet delivery channel is described in detail. It is shown how balance inquiries, electronic bill payments, transaction details, funds transfers, automatic reconciliation, and customized reports are all enabled for various bank products, such as checking accounts and savings accounts, all based on the bank's business requirements and taking careful consideration of the bank's technology constraints. Both the physical and logical architectures for the Internet banking channel are detailed and diagrammed. These architectures resolve two primary bank requirements: (1) a complete security architecture that addresses all of the potential securities threats incumbent with doing business on the Internet, and (2) integration with and reusability of the bank's transaction processing mainframe systems and middleware. There is special focus on the integration required with these backend systems and associated middleware, and on the technology used to accomplish this integration. The non-technology aspects of this project to create this on-line banking channel are also described, in particular the various methodologies, project roles, and lessons learned are outlined. This paper serves as a compilation of what we learned through our experiences in enabling this bank to interact with its customers over the Internet.

The corresponding architecture slides for this paper are here.

1 Introduction

Both the structure and the content of retail banking have begun to evolve in recent years, necessitated by competitive market pressures and facilitated by easing of government regulations and advances in computing and other technologies. The traditional "bricks and mortar" branch bank has become too expensive, inefficient, and restrictive to act, for the industry, as the sole storefront to its customers in the emerging re-engineered financial services marketplace of the mid to late 1990s. This structure is slowly being supplemented by banks with newer, more efficient, and more cost-effective delivery channels, such as the Internet and Worldwide Web. Riegle-Neal has made it easier for banks to use these new delivery channels to cross state boundaries and reach new customers digitally. These newer "self-service" delivery channels can help banks in their quest to deliver any products through any delivery channel at any time of the day or night, thereby maximizing their potential revenue stream as well as satisfying, retaining, and perhaps growing their customer base through excellent customer service and cross-selling of new products with that old-fashioned personal touch, that seems to have been lost somewhere along the way.

This paper describes a real world project whereby the Hewlett-Packard Company Professional Services Organization (PSO) helped a large U.S. retail bank build a new Internet-based delivery channel. Specifically, the first phase of the project, which was implemented in late 1996 and early 1997, is discussed in detail.[1] This new Internet delivery channel will be used by the bank to deliver products and services to its customers as well as to cross-sell new products in a secure, personal, "just in time" manner.

First, the client bank is profiled, and its business objectives are summarized. In the second section, there is a summary of the methodology that HP used to elicit and document project principles, specific business requirements, and technology and infrastructure constraints. This process was not occurring in a vacuum, and as such, some of the internal corporate dynamics are discussed. Also, in this section, the various roles that played a part in the project are discussed. Following this, the Internet banking application functionality is briefly described. In the next section, the physical and logical architectures for the solution are overviewed and diagrammed. Special mention is made in two primary areas: (1) it is shown how the architecture for this Internet banking solution addresses the primary security threats on the Internet, and (2) it is shown how this architecture integrates with the backend bank transaction processing systems. Next, the key partners that the HP team collaborated with during this project are discussed. Finally, the last section summarizes recommendations and conclusions based on lessons that were learned during this project.

[1] At the time of the writing of this paper, the second phase of the project is being implemented.

2 Bank Business Objectives

The client bank, like many banks, has a strong business objective to lower the cost for each customer transaction. Their branch based transactions are the highest by far,. followed by the call center, the ATM, and then the voice response unit (VRU). Self-service transactions over the Internet are estimated to be *more than one hundred times less costly* to deliver than branch based transactions that involve bank personnel. One of the measures that the bank is taking to reduce this overhead is to sell off and/or consolidate a large percentage of their branches.

Another critical bank objective is the retention of their loyal banking customers across the country. Addressing customer retention has several factors -- beating the competition, avoiding disintermediation, and providing superb customer satisfaction. There is the threat of competition from both regional and national banks. A regional banking competitor, for example, had already introduced their Internet banking package earlier in 1996, and our client bank had an urgent desire to also get to market as quickly as possible. On the not so distant horizon, there is also the threat of disintermediation by other financial services companies as well as by non-traditional competitors, like software publishers. If the bank were to be disintermediated, it would lose the customer relationship -- the very thing that it cherishes most. And to control the relationship with the customer, the bank has to control the delivery channel.

Probably the most important factor to retaining customers is delivering satisfaction through convenience and value. As one example of convenience, our client's customers were literally begging the bank for Internet banking. They were sending electronic mail over the bank's web site asking when they would be able to start banking from home over their personal computer. They were asking this due to the convenience of when, where, and how they perform their banking. They were also asking this due to the control that a self-service banking channel provides to the bank customer. Being able to perform electronic bill payments and fund transfers when one absolutely *needs* to, for example, gives a customer an enormous feeling of control and satisfaction -- which is essential for customer retention.

The other part of the customer satisfaction equation is providing value by delivering meaningful services to customers, and delivering these in a personal manner. Here, delivering meaningful services implies more than simply showing product rates or webifying checking and savings products transactions -- functions that are already available through other delivery channels. Rather, it means creating new services that are most appropriately delivered through the new channel and that extend the bank's existing products and services and that have meaning to the bank's customers. In addition to the electronic billpay service, which is also available on the bank's VRU, the bank wants to provide a brokerage service, where equities can be researched and bought and sold. One of the risks, however, of using an automated delivery channel to provide these new services is the potential for relatively little or impersonal interaction. Therefore, the bank has an object for providing a personal customer experience as the backdrop for this delivery channel. This implies providing as part of the overall customer experience, individualized web pages, individualized cross-sell marketing

messages, opportunities to interact with the bank's customer service representatives -- and potentially other bank customers, and in the future, individualized products. The cross-selling marketing messages are an area that need to be handled sensitively, so as not to alienate customers with obtrusive sales pitches. Rather, the bank would like to deliver these in a personal, value-added manner by having the messages triggered by personal life events. For example, a child turning eighteen could trigger a marketing message offering a special college loan or checking account. In this way, instead of being perceived as an unsolicited intrusion, the messages would hopefully be perceived as extensions of an already personal relationship between the bank and the customer.

3 Project Startup Methodology

3.1 Internet Banking Principles

Part of HP's methodology in the beginning of the project to gain momentum was an executive workshop, whereby in a three hour period, key executive stakeholders were organized so that project principles, themes, and requirements are elicited. First, everyone agreed to a short list of project principles that were so unifying and global that they guided the development of the remainder of the business requirements. The project principles that were gathered during this workshop are:

1. *Strong bank branding.*
2. *Unique Value to customers.*
3. *Customer centric -- reflecting the customer relationship.*
4. *Must be easy to use and intuitive to the customer.*
5. *Finally, and most important, it must be secure!*

Next, the workshop facilitator encouraged all stakeholders to brainstorm on the various business requirements for the project. The facilitator then worked with the participants to gain consensus on which of these were absolutely essential for the various phases of the project, and which of these would be "value-added" functionality. The project team also provided its input, based on its knowledge and understanding of the Internet and related technologies and where and how they are evolving and maturing. Coming out of this workshop, then, was a set of unifying project principles and a list of prioritized requirements to which everyone agreed -- the bank business personnel, the bank technology personnel, and the project team[2].

The next step with which HP proposed to work with the bank was a rapid application prototyping workshop, whereby HP would work with the bank's retail business unit stakeholders to understand the exact functionality, branding, and user experience requirements for the Internet banking project. Here, for each unit of functionality that the bank required, HP would *rapidly develop*[3] and gain consensus with the stakeholders

---

[2] It is important to note here that the project team was composed of both Hewlett-Packard consultants as well as bank employees. The exact composition of this project team and their roles are detailed later in the next section of this paper.

[3] HP uses an HP rapid application development (RAD) methodology for implementing Internet banking projects. This RAD process differs from traditional software development projects in that it is timeboxed and iterative in nature. Each development cycle is limited to approximately one calendar quarter. In this unique manner, new features and functionality for our clients' Internet bank can be introduced on a regular and frequent basis -- *in web time.*

on the corresponding look and feel for the user interface as well as its navigation. Of importance here to the bank is the bank branding as well as the customer experience. Due to the timeline on which the bank was proceeding on this project, it opted instead, in this first phase[4], to do a full Internet banking proof of concept.

3.2 Project Requirements and Constraints

Some of the functionality requirements that were gathered during the workshop are:

1. *Checking and Savings Account products to be supported.*
2. *Electronic Bill Payments.*
3. *Customer Defined Accounting and Transactions.*
4. *Real-time Account Balances.*
5. *Funds Transfer between Customer Accounts.*
6. *Some value-added functionality for differentiation from the competition.*

It was also determined that the second phase of this project would support the following requirements:

1. *Customer enrollment (e.g., Account Openings).*
2. *Credit card product supported.*
3. *Integration with customer service department.*

These project principles and functionality requirements had to then be mapped to the real world by taking into consideration the existing technology infrastructure of the bank. This infrastructure includes:

- the various bank mainframes,
- the banking applications and their corresponding databases that support the various bank products,
- the transaction processing operational software,
- the transaction programs written by the bank to support a multitude of banking functions,
- middleware purchased and customized by a third party for billpay through the VRU,
- the existing billpay provider,
- the SNA gateway,
- the customer data warehouse,
- the Internet architecture designed by the bank,
- the various security mechanisms in place,
- and the various business rules employed by the bank applications.

The process that the project team used to uncover all of this information were detailed interview sessions with the bank information systems (IS) technology stakeholders. In

[4] In the second phase, HP did conduct this rapid prototyping workshop with the bank.

[5] The current version of the VBM software now does support real-time processing.

particular, several key technology infrastructure workshops were scheduled on the following subject areas to elicit technology requirements and constraints:

- Security
- Internet Infrastructure
- Legacy Data Integration
- Application Architecture

As a result of this process, the following technology requirements and constraints were identified:

1. The Internet banking project needed to reuse the existing mainframe infrastructure.
2. The project needed to reuse the existing home banking middleware.
3. The project needed to use the same billpay provider as the telephone banking system.
4. The bank had very little Internet background.
5. The bank had very little bandwidth to assist HP.
6. Real-time data access to mainframe legacy data was mandatory. (This required modification to the shrink-wrapped web application, that would be used.)
7. The bank had certain preferences for how legacy data was to be retrieved, however, much of this mechanism had never been tested or integrated.

3.3 Alternatives, Gaps, and Solution

The project business and technology requirements that we gathered, along with the overall project principles that guided the project, lead to several possible solution alternatives for our client bank. Two of these were based on third party application software, and one was a custom software development option. Each solution that was considered had certain gaps that needed to be filled to meet the requirements. Based on the direction that we received from our client, we decided to implement an Internet banking solution based on the Virtual Bank Manager (VBM) Internet banking application software from Security First Technologies (S-1). One of the gaps for the solution that was selected was the fact that the version of the software that we were using did not, at the time, support real-time processing,[5] which was one of our client's business requirements.

3.4 Other Dynamics

The backdrop for this Internet banking project was the healthy skepticism by several bank executives about the security of conducting retail banking on the public Internet. Another project, one focused on delivering PC home banking services, was using a private network linked to the bank to accommodate its dial-up customers, an infrastructure which was perceived as more secure than the Internet. Hewlett-Packard devoted much effort to work with these executives on their skepticism. HP put together presentations, educational seminars, and solution partner meetings to describe in detail the exact nature of the various security threats over the Internet and how they could be alleviated. It was show how with various technology solutions and security architectures these security threats could be addressed. We had previously proposed an

in-depth security assessment and security architecture engagement to the bank, however due to the timeline on which our client was proceeding, they opted instead to use HP as the primary integrator for aa full Internet banking proof of concept.
Also, the accelerating popularity of the Internet had caused our client to rethink some of their original requirements -- specifically to rethink the model of a proprietary dialup network providing only a single home banking service to a standards-based dialtone providing a agglomeration of services[6].

3.5 Project Team Roles

There were many important roles that played a part throughout the project. Some of these roles, such as project manager and lead architect, were necessary for the duration of the project. Other roles, such as transaction processing expert and technical writer, were only required for a part of the project. These roles are summarized below.

- *Project Sponsor*
- *Business Project Managers*
- *Lead Architect*
- *Business Liaison*
- *Meeting/Personnel/Logistics Manager*
- *Technical Project Manager*
- *Application Expert*
- *Process Methodology Expert*
- *Facilitator*
- *Mainframe Infrastructure Expert*
- *Legacy Data Expert*
- *Infrastructure Architect*
- *Security Architect*
- *Transaction Processing Expert*
- *Programmers*
- *System Administrator*
- *Technical Writer*
- *Account Manager*

In some areas, a single role was filled by multiple persons, while in other areas multiple roles were filled by a single person. HP PSO project personnel were used to assume most of these roles while our client's project personnel assumed the roles of Project Sponsor, Business Liaison, Legacy Data Expert, and Meeting/Personnel/Logistics Manager.

---

[6] Other advantages of the Internet web browser approach include (1) no investment in client software, (2) no software distribution, (3) ability to be innovative without changing current technology or sending out new releases, (4) providing central management of the direct banking application, and (5) providing a direct relationship with the customer.

4 Application Functionality Overview

Virtual Bank Manager (VBM) is the foundation for Security First Technologies' Internet banking application software. It supplies standard bank offerings such as account openings, demand deposit accounts, electronic bill payment capability, and check ordering. Credit cards, brokerage, and insurance products are also available as options.

With VBM, customers have round-the-clock access to their accounts via the bank's "home page," their lobby on the World Wide Web. The customer interface is intuitive, mouse-driven, and user-friendly, giving customers:

- Customer sign-on and authentication
- On-line account statements, balance inquiry, reports, and register views
- Automatic statement reconciliation
- Transaction categorization
- Funds transfer between accounts
- Electronic bill payment capabilities
- Account transfers
- Tutorial and On-line Help for new customers
- Information server with product demo for potential customers
- Direct e-mail link to the bank's customer service department
- .QIF Export format (Quicken/MS Money)

Virtual Bank Manager supports checking, money market accounts, joint accounts, bill payment, CDs, ATM/Debit cards, credit cards, check imaging, and traditional paper checks. Customers who access VBM on-line can review account activity, enter transactions into an on-line account register, pay bills electronically, and print reports to indicate income and spending trends.

With VBM, new customers may apply to open an account with the bank over the Internet by completing an account application. The customer is required to submit the application on-line, print the application form along with all necessary and applicable disclosure statements, sign the application form and the disclosure statements, and mail the documents to the bank, along with funds for the customer's initial deposit.

Bank customers can view an electronic account register that is similar to a traditional checkbook register. Each time a customer writes a check or enters a deposit, the customer may enter the transaction into an electronic register if they wish to have a completely up-to-the-minute bank balance. The account register displays both cleared and uncleared transactions. When those transactions clear, VBM matches them with the register entries. If there is a difference, VBM notifies the customer about the discrepancy. If the customer prefers not to manually enter transactions, they can wait until the cleared transactions are posted to their register. Both the account statement and the account register provide for payee/payer and category fields so customers can itemize income and expense categories.

After a customer's items have cleared, the item appears in both a customer's account register and the customer's account statement. The on-line account statement is an electronic version of the statement a customer would receive in the mail from a traditional bank. Unlike a traditional monthly statement, this electronic can be accessed any time at the customer's convenience. The customer can adjust the dates of the statement and go back as far as eighteen months.

Customers are able to send electronic payments to any payee. They can also select to have bills paid at a predetermined frequency (i.e. monthly, quarterly) without additional input. Customers can stop payment on any electronic bill payment or automatic payment by deleting the payment from a pending payment list or by unselecting an automatic payment for a specific payee.

Electronic bill payments that are initiated with VBM are subject to customer-defined limits and bank-defined limits. These limits protect the exposure that a customer has to any fraudulent activity. They are also incorporated into log files to automatically detect potentially fraudulent activity.

At the time of application, a customer is asked to set *customer-defined limits* on their accounts for electronically initiated bill payments. These limits can be set on a transaction basis and on an account basis. A transaction-based limit would limit the dollar amount of any single payment. An account-based limit would limit the total amount of payments that could be made in any one day. These limits and actions can be changed periodically by the customer.

Bank-defined limits are limits maintained by VBM and affect all accounts, overriding customer-defined limits. There are three types of bank-defined limits: transaction-based, account-based, and payee-based. Transaction-based limits and account-based limits are the same as those defined in the customer-defined limits section. Payee-based limits are limits on payments to any one payee across all accounts.

5 Solution Architectures

The solution that the HP project team implemented, that met the stated functionality requirements and technology constraints, is composed of data processing scenarios, and a variety of hardware and software components. These span the path from the bank customer web browser running on a PC over the public Internet to the bank servers and legacy mainframe systems. Several views of the solution are described below, starting with the processing flows and their modes of operation. Following this, the physical architecture highlights the hardware systems and high level networking protocols that are used. The logical architecture is shown in the next section, and it focuses on the software components and also the more detailed protocols and application programming interfaces (APIs). Finally, the security architecture is discussed, paying particular attention to how various security threats are addressed.

5.1 Data Processing Flows

VBM consists of a series of graphical web pages with which the customer interacts and a software program that executes customer requests. When customers wish to view their account balances or perform transactions, VBM communicates with an Informix relational database which stores account information for each customer and batches up transactions that need to be communicated with the bank's demand deposit account (DDA) software at the end of the day (or however often the bank wishes this to take place.)

There are six (6) sets of processing flows that will need to occur in order to support Internet banking to our client's customers with the VBM software. These flows are introduced below.

1. Initial Customer, Account, and Relations Setup Processing

This processing loads existing and new customers and their accounts into the VBM database. It also relates customers to their accounts.

2. Initial Bill Payee Setup Processing

This processing loads merchant information and recurring payments for each existing customer from the bill payment system into the VBM database.

3. Account Detail (e.g., debits, credits, reversals)

This daily processing loads daily customer transaction activity (e.g., all debits, credits, and reversals) from the client's DDA into VBM, thus providing account detail information for each customer.

4. Billpay Processing

This daily processing verifies that a customer's pending electronic bill payment for that date has sufficient funds in order to be processed. If so, that debit is memo-posted to the customer's account, and the master billpay account for that bank is memo-posted with the credit. Also, the electronic billpay is appended to a batch file which is transmitted to the mainframe for later processing by the bill payment vendor. Also, new billpay merchant information (e.g., merchant adds and merchant edits) from VBM are transmitted to the bill payment system on the mainframe.

5. Funds Transfer Processing

This processing transfers funds from one customer account into another account for that same customer, by loading funds transfer requests from VBM into the mainframe DDA, using memo-posted debits and credits. As in billpay processing, a sufficient funds verification is first performed. An email is automatically sent to the customer documenting the success or failure of the operation.

6. Account Balance Processing

This processing occurs when the customer logs into VBM so that the latest customer account balances are displayed. For each customer account, the mainframe is queried for any memo-posted debits and/or credits and the resulting customer balance is displayed in their web browser.

Each processing flow requires some data exchange between the mainframe system and VBM. The data exchange can occur in three different modes outlined below.

a) Batch: automated off-line processing in which common data files are exchanged.
b) Near-time: automated on-line processing in which messages are exchanged.
c) Real-time: automated on-line processing in which messages are exchanged, triggered by a customer event (e.g., customer login).

Batch Mode:

In batch mode, each processing flow spans execution across the client's mainframes and the VBM server. In general, each will have the following steps associated with it:

A. *Extract* data from source database into common data file.
B. Optionally, *transfer* this file to the destination host.)
C. *Load/Update* data from this file into the destination database.

As such, each batch processing flow that spans execution from our client's mainframes and the VBM server is composed of at least two (2) associated batch job steps. Some processing flows, such as billpay, will require additional batch job steps due to round-trip processing.

When loading/updating and extracting data between the VBM database and the common data file, embedded SQL is used. When loading and extracting data between the mainframe and the common data file, out client's existing middleware is used.

Batch mode processing is typically used for daily account detail and billpay processing, as well as initial customer, account, and billpay setup processing.

Near-time Mode:

Near-time Mode is an improvement on batch processing, in that instead of exchanging data via common data files, messages are exchanged in *near real time* (thus the name). This mode is used to improve the processing flows that are time critical, such as electronic funds transfer. In this way, instead of waiting a potentially considerable amount of time for a response via a batch file, and thus for a processing flow to complete, a message exchange occurs in near real time. This message exchange is initiated using frequent scheduled processing times (e.g., every five minutes). In the example of funds transfer, then, a funds transfer request is memo-posted within five minutes of the request, thus allowing a later ATM withdrawal by the customer to succeed.

Real-time Mode:

The major difference between near-time and real-time modes, is that near-time processing is not triggered by a customer action. Real-time processing mode *is* triggered by a customer action. In this mode, data exchange occurs in the most timely fashion. The real-time processing mode is used to display actual customer account balances from our client's mainframe, when the customer logs into their account. ATM withdrawals, for example, that could have been recently memo-posted would be accounted for in this real-time account balance.

Our Implementation:

These various processing modes complement each other in the implementation for our client. The current VBM software, at the time of our implementation, did not support real-time processing. Until this time, batch and near-time processing are used for most of the processing flows, above. The HP team decided that, in order to be successful on this project, we needed to customize the S-1 VBM software to be able to meet our client's business requirement for real time access to account balances.

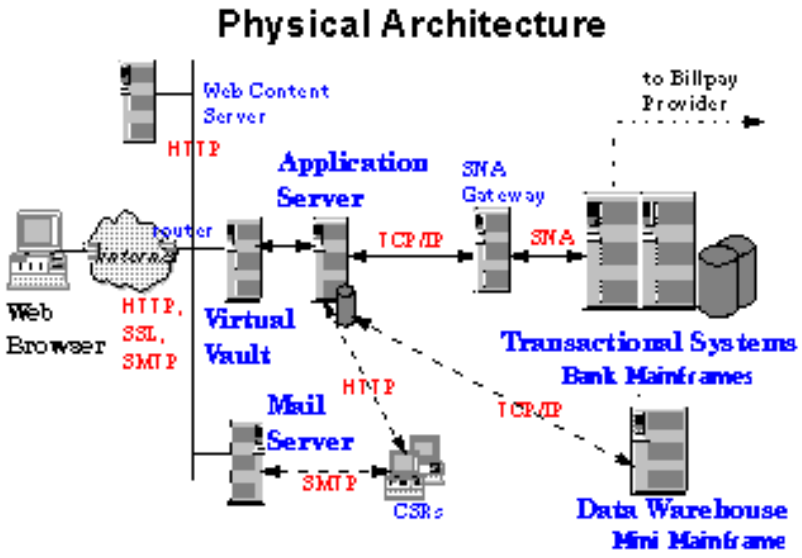
In our implementation, regardless of the data processing flow used, our client's existing CICS transaction programs and a new HP object layer will always be used. For example, instead of writing directly to the VBM batch file layouts in a batch processing scenario that will be migrated to real-time in a later phase of the project, we decided to create a middle layer of objects that model our client's banking transactions and that encapsulate all of this data access. In this way, we are not coding directly to the particular data processing mode, and as such there will be very little throwaway code. Specifically, when the newer VBM release supports real-time processing, this middle layer of banking transaction calls and logic are simply moved from the batch adapter to the real-time adapter and the file I/O is simply removed.

It was also suggested by the HP team, that even when real-time processing is supported, it will not be used exclusively in all processing modes. Rather it will be used in tandem with batch and near-time processing. Specifically, some of the processing lends itself naturally to batch processing (i.e., initial customer/account/billpay setup and daily customer transaction detail processing). And batch mode processing can occur in an off-line batch mode at uncongested customer traffic times to increase the overall performance of our client's Internet bank. For example, instead of downloading *all* of the DDA transactions since the last customer signon -- which could be quite large -- in real-time when that customer next signs on, it will be better from a performance perspective to download *only the incremental* DDA transactions since the last daily batch download.

For this phase of the project, we decided that batch processing is used for account detail and initial customer load. Near-time processing is used for funds transfer and billpay. Finally, real-time processing is used for querying customer account balances, in the new VBM real-time interface that we developed for our client.

5.2 Physical Architecture

There are several components in place that comprise the physical infrastructure for this solution. The high level protocols are in red, the physical hardware components installed at the bank are in blue. The infrastructure in place outside of the bank, that is used in the solution is in black.



Web Browser

This is the bank customer's PC, at either home or work, running a web browser, such as Netscape Navigator or Microsoft Internet Explorer. The only required software component for the bank customer is off the shelf software -- quite a contrast to the proprietary home banking model, where a custom developed piece of client software is required. This is important to the bank, because no longer does the bank have to be in the software distribution business, which is not really its core competency.

Internet

The only other component required for the customer is to have access to the Internet via an Internet Service Provider. From the bank's point of view, a connection or set of connections to the Internet is also required so that it may provide services to its customers. However, when compared to some proprietary dialup home banking solutions, the bank does not have to be in the business of managing large banks of modems and telephone lines -- again, not really its core competency.

Filtering Router

The filtering router is responsible for connecting the bank to the Internet, as well as for providing an initial layer of security.

Mail Server

The Mail Server is a server computer that acts as the conduit which routes electronic mail between the customer and the bank's customer service representatives (CSRs). For example, the customer may have questions or problems that are most conveniently communicated through email. Also, email is sent to customers by CSRs when problems are resolved, and by bank business managers as a conduit for marketing information. Finally, when important events (e.g., transfer of funds successful) are registered by the Internet bank application, email is also sent. This is an HP Unix server computer, which runs both a mail server, a firewall package, as well as the Domain Name Service (DNS).

Web Content Server

This web server is the repository for all web data, other than those provided through interaction with the Internet banking application. For example, the bank's home page and marketing information are stored on this server, as well as any other static HTML, graphics, etc. This is an HP Unix server computer running Netscape Enterprise Server.

Virtual Vault

The Virtual Vault is an HP server computer that acts as the single point through which all *secure* Internet banking transactions occur. As such, it acts as a firewall that also provides the necessary security to prevent and audit known and unknown security threats. Since all of the information and monetary assets that the bank owns are now attached to the public Internet composed of tens of millions of computers, the Virtual Vault is responsible for protecting the bank's exposure. In particular, it protects all of the Internet banking applications and data against security threats by either external or internal hackers.

Application Server

The server portion of the Internet banking application runs on this HP-UX based application server, along with the data repository for the Internet banking information. The application is Security First Technologies' (S-1) Virtual Bank Manager (VBM). The data repository is an Informix relational database. The types of information held in various Informix tables include the following:

- *Customer information*
- *Customer Payees*
- *Transactions*
- *Transaction types*
- *Customer Accounts*
- *Categories*
- *Pending payments*

It is important to note that this information supplements the information already maintained on the mainframe transactional systems. For example, transaction categories and Internet-specific information are store on this application server. This application server also acts as a data warehouse, providing in some cases more historical data (e.g., sixteen months of customer transaction information, for example, for tax reporting purposes) than the mainframe provides.

SNA Gateway

The SNA gateway is an IBM computer running the AIX Unix operating system. It is a protocol converter routing TCP/IP application requests into SNA CICS transactions.

Transactional Systems

The transactional systems are those systems upon which the core banking applications reside. These are a private nationwide network of IBM mainframes running the MVS operating system and CICS transaction processing monitor. The primary mainframe applications that are used are the Hogan Systems Demand Deposit Account (DDA), the Braun-Simmons Interpose billpay system, as well as internally developed CICS middleware using various VSAM files as information repositories, mostly written in COBOL.

Billpay Provider

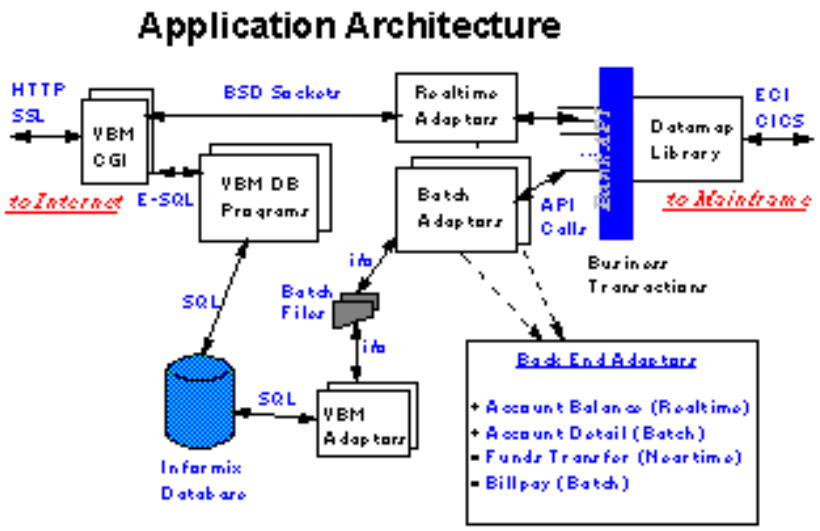
The billpay provider is a third party company that is responsible for providing the electronic billpay service. Customers request to make payments to their merchants with whom they do business, and the billpay provider acts as the conduit payment provider. When a billpay request is made, the customer uniquely identifies the party to be paid with an address, account number, etc. Funds are then transferred from the customer's account to the billpay provider's account. Payment is then made, preferably electronically, and if this is not available, then a paper check is cut on behalf of the customer. Customers control when the payment is to be made, and thus when it is debited from their account. Also, either recurring or one-time payments can be requested. Physically, a dedicated and secure private connection is required between the billpay provider and the bank's transactional systems, so that the electronic bill payment data can be transferred. As stated previously, the relationship with this third party and the physical connection was already in place to serve the VRU channel.

Data Warehouse

The data warehouse is a repository of historical information on the bank's customers as well as information on the bank products. It resides on an IBM mini-mainframe running the MVS operating system, with the data residing in a DB2 relational database. Personal life event information is extracted from this data warehouse and presented to the customer in the web application, so that products can be marketed in a personal, relationship-oriented, non-intrusive manner.

5.3 Logical Architecture

The software component, protocols, and APIs that comprise the logical architecture of the Internet banking solution are discussed below.



HTTP

HyperText Transfer Protocol (HTTP) is the Internet protocol that is used to send data between the bank customer's web browser and the bank web server. HTTP, by itself, is a non-secure protocol, and must be used in conjunction with other mechanisms to establish security. HTTP, therefore, is used for transferring non-secure marketing information between the customer web browser and the web content server (see above).

SSL

Without adequate security measures, both the bank and its customers have to worry about electronic fraud. This includes data tampering, eavesdropping, and forgery. Secure Sockets Layer (SSL) is the component in this architecture that prevents these types of fraud by securing web sessions between the bank and the customer. Public key certificates are used to encrypt important customer data and to authenticate the bank and optionally the customer. SSL allows sensitive information (e.g., account information, passwords) to be shared between browser and server, yet remain inaccessible to third parties. It also ensures that data exchanged between browser and server cannot be corrupted -- accidentally or deliberately -- without detection. In particular, SSL secures information between the customer web browser and the Virtual Vault. SSL also allow the web browser and web server to authenticate one another by exchanging public key certificates.

VBM CGIs

Common Gateway Interfaces (CGIs) are the application programs that run on the web server -- in this solution, the Virtual Vault -- to implement some functionality. VBM CGIs are the client side of the S-1 Internet banking application software, that make E-SQL calls on behalf of some customer request over the web.

E-SQL, VBM DB Programs, and Informix Database

E-SQL is Informix's embedded SQL technology that is used by the VBM CGIs to query and update customer, account, and billpay data in the Informix database from across a network. In particular, the VBM client side CGIs on the Virtual Vault invoke some customer requested application functionality by connecting with the server side VBM database programs on the Application Server using Informix communication libraries. The VBM database programs make embedded SQL calls to the Informix relational database to query and modify the necessary data.

VBM Adapters and Batch Files

The VBM data adapters are another component of the S-1 VBM application software. They supply off the shelf functionality whereby customer, account, and electronic billpay data can be both retrieved from and updated into the VBM Informix database, without having to know the schema of this database and without having to write any SQL. Some of the data flows that have corresponding VBM adapters include:

Customer and Account Maintenance Information
Add Account Detail
Get Transfer Funds Requests
Get Billpay Requests

These adapters are processed on the Application Server and will either extract from or load into the VBM Informix database using human readable text files that are processed by the legacy data backend adapters (see below).

Backend Adapters

The backend adapters are software developed by Hewlett-Packard for our client that interface with both the VBM Informix database and the bank's back end systems. The adapters are scheduled to run in either batch or near-time, as directed by bank operations personnel. There are backend adapters for the following data flows:

Customer and Account Maintenance
Account Detail
Funds Transfers
Electronic Billpay

Typically, the customer and account maintenance, account detail, and billpays are scheduled to run as nightly batch jobs, and the funds transfers are executed in near-time, meaning in almost real time in response to a customer request. When they do execute, they read from and update bank mainframe data directly in real-time using a synchronous messaging interface described below. The adapters are abstracted from having to know any detail about this mainframe by encapsulating all legacy data access with layers of software called *Client Bank API* and *Datamap Library* (also described below).

Real-time Adapters and BSD Sockets

The components described above comprise one series of data flows between the customer web browser and the bank mainframe systems. However, our client had a technical requirement to do real-time access to their backend legacy systems. Specifically, their requirement was to access latest customer account balances, including any ATM or other activity that may have been performed that very day. The Internet banking application did not yet support this real-time functionality, so the HP team worked with our client and S-1 Technologies, and modified the VBM source code to include this real-time account balance functionality. Specifically, we modified the C++ source code for the *Account Balance* CGI so that instead of accessing the Informix database for account balances, the CGI made a networking connection (using a Berkeley sockets mechanism) to middleware that would then access the real-time account balances (see below for more detail).

Client Bank API, Data Access Library, ECI

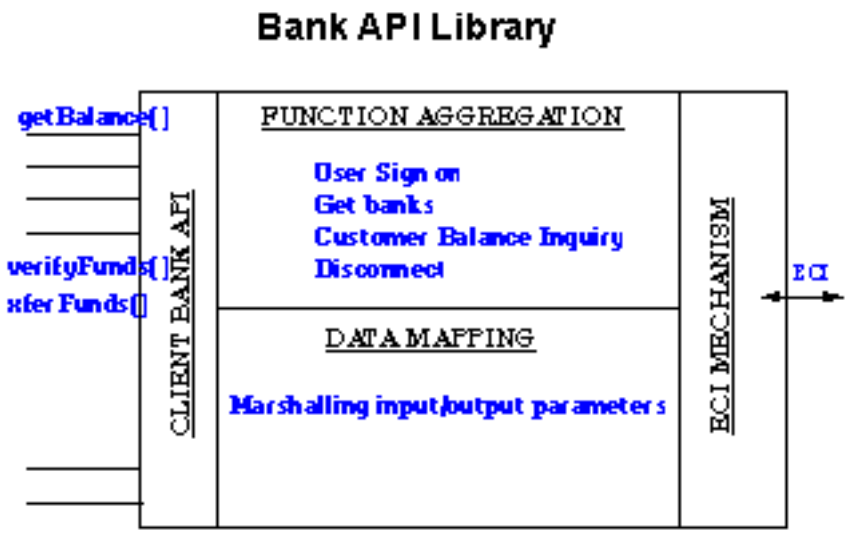
Rather than have the application CGI connect to the bank mainframe directly, the project team made the architectural decision to access the real-time account balances using a piece of middleware that was developed for this project. Reasons for this include:

- Scalability
- Reusability
- Portability

Specifically, an architecture that provided two-tier access to legacy data would not be the optimal solution for simultaneous access by a subset of the bank's customers. A three-tier model provides a better model by offloading the data access requests to a second tier. Secondly, if the data access and business logic was coded directly into the CGI, then any change to the logic would require changes to the CGI programs. Eventually, these CGIs would become part of the baseline VBM application, so we opted to embed the logic in the second tier. Finally, this data access layer could be used by other projects ongoing in the bank that needed a way to get to this mainframe customer account data from an application not residing on a mainframe.

The solution that we designed is a data access library that is designed in an object-oriented manner. It has several components:

- A client bank business API
- A function aggregation and data mapping component
- A component responsible for implementing and abstracting the legacy data access mechanism -- in this case IBM's External Call Interface (ECI).



The client bank business API is the interface that is used by any other application or object that needs to use these data access services. It abstracts all of the implementation details into a very simple and intuitive set of calls, such as:

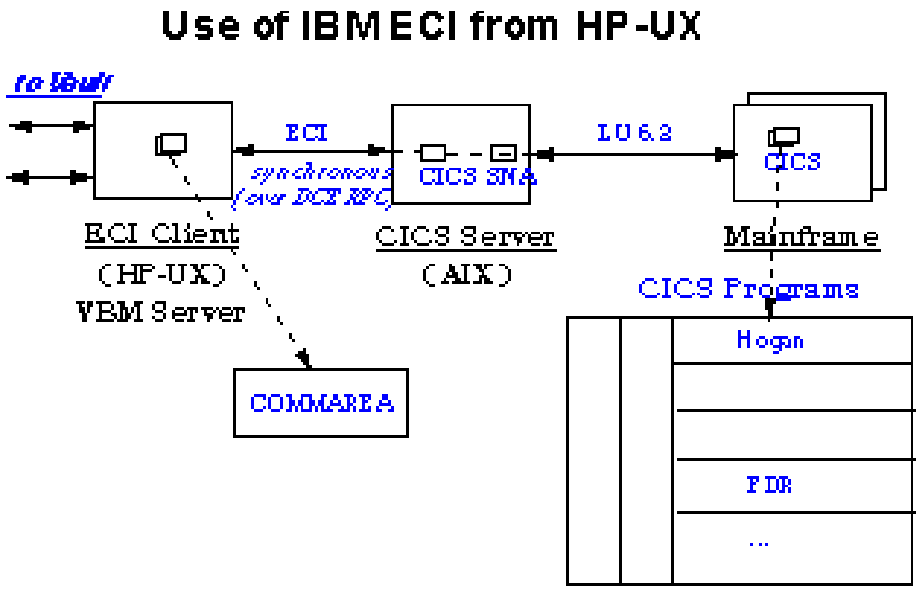
⇒ Get Balance
⇒ Verify Funds
⇒ Transfer Funds

These are the set of transactions that a customer needs in order to do business with the bank. Amazingly enough, these transactions, while they did exist, were not implemented using an easily accessible, documented, and clean interface. Further, there were a set of other supporting transactions that were required, before the primary banking transaction was invoked.

Thus, our data access object performed function aggregation so that only one transaction was required. It also implemented the necessary data mapping that was necessary for marshaling the transactions input and output parameters to and from the communications area (COMMAREA) and between each supporting transaction call.

Finally, due to two of the customer's requirement, we used the IBM ECI interface to access the customer account data hosted on the bank's mainframes in various VSAM files. Remember, that these requirements were to reuse existing mainframe infrastructure and that they had preferences to use ECI, even though it had not been fully tested when used in this manner.
ECI allows software programs that are not CICS based to implement existing CICS transactions without any modification to those transaction. This reuse of their CICS software was extremely important to our client and this is ultimately why we went with this mechanism, which turned out to be a very nice way to access this legacy data. ECI, which is implemented on top of Distributed Computing Environment (DCE) remote

procedure calls (RPCs),. uses a simple send/receive metaphor for data access through a communications area.



Finally, note in the application architecture diagram above, that this data access library is used by both the real-time and the batch adapters. In this way, all data access throughout the system is isolated to one software object.

5.4 Security Architecture

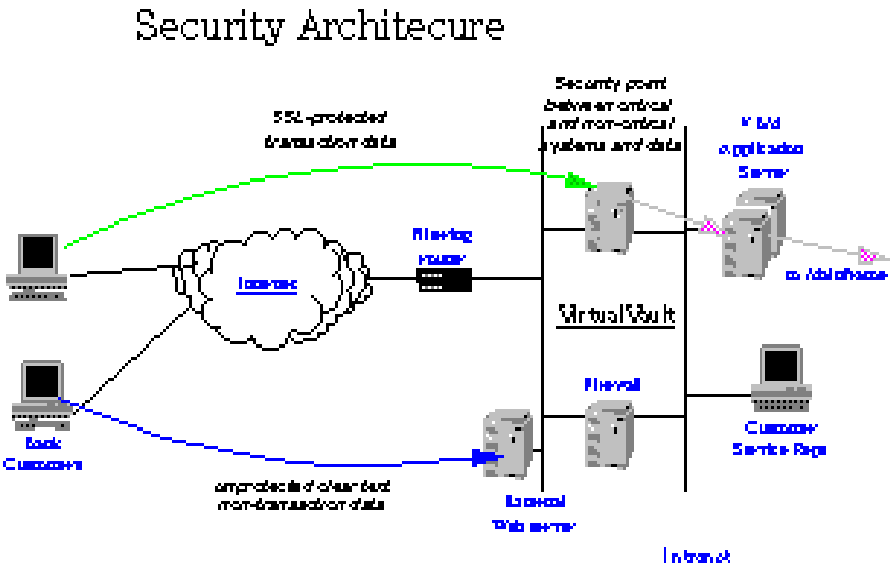
The various security threats that exist on the public Internet needed to be accounted for in the solution architecture for our client's Internet bank. Some of these threats are:

1) Fraudulence
   - *User spoofing*
   - *IP spoofing*
   - *DNS spoofing*
2) Eavesdropping / Disclosure
3) Transaction Integrity
   - *Modification*
   - *Insertion*
4) Attacking bank infrastructure
   - *Attacking web server and firewall*
   - *Attacking web application*
   - *Collusion of insider with outside attacker*
5) Trojan horses
6) Taking over bank services
   - *Denial of service*
   - *Re-directing services*
7) Inadvertant threats
   - *Web server instability*
   - *Application instability*
   - *Incorrect software configuration by Operations*
   - *Web server security problems*

These threats can be viewed as four general categories of security attacks:

I. Unauthorized access to network transactions
II. Unauthorized access by outsiders to server data
III. Unauthorized access by insiders to server data
IV. Attacking client and server application integrity

The security architecture that is show below solves every one of these attacks -- except for attacking client PC application integrity, which is out of the control of the bank.



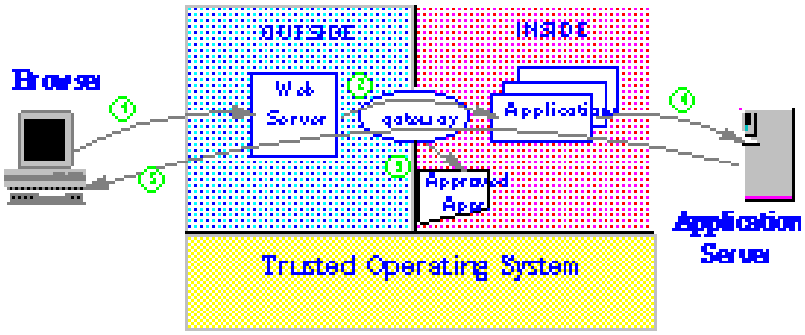
The filtering router is programmed with rules that determine what gets through. Typically, only HTTP, SSL, and SMTP are allowed only over well-known ports. Also, source route attacks and inside IP address spoofs are stopped here, as well as attempts to reprogram the router.

The Secure Socket Layer (SSL) technology inherent in today's web browsers and web servers encrypts banking transactions so that confidential information like account numbers and account balances all remain private to the customer. Further, SSL checks for integrity of the banking transactions, so that no modification is possible. Finally, the banking server is authenticated so that the chance of a renegade fraudulent server pretending to be the real bank server is not possible.[7]

Virtual Vault is the one of the most important components in the security architecture that prohibits both insiders and outsiders from gaining access to our client's monetary assets.

[7] Client-side authentication and support of smart cards is also possible with SSL, possible in a future phase of the project.

Virtual Vault



A commercialization of U.S. government-deployed (B1) trusted operating system software, Virtual Vault acts as a military strength firewall. It provides much greater security, however, than standard operating systems through employment of *least privilege* and *information separation* mechanisms. Least privilege addressees super-user weakness by substituting a privilege check for each super-user check in the operating system. Applications like web servers, then, are only granted the specific privileges needed for the task at hand, while eliminating super-user from the system.

Information separation allows for information assets to be kept inside inaccessible compartments while the web server and site presentation is relegated to an outside compartment. Specifically, the web server runs in the "outside compartment"; it is here that the HTML is displayed and user input is collected. The VBM Internet banking application runs on the protected "inside compartment" of the Virtual Vault as a collection of Common Gateway Interface (CGI) applications, while the HTML. Before each CGI is run,a check is made to see if it is a valid software program that is part of the Internet banking application. Further, the checksum for each CGI is checked to make sure that it has not been modified or replaced.

Together, these security features prevent tampering and, hence, provide the extra assurance that hackers will not be able to compromise the bank. Because of Virtual Vault's security features, even inadvertant threats -- such as software bugs in the web server or the Internet banking application, or even incorrect software configuration by Operations personnel -- are thwarted. This allows much greater confidence by our client in the security of their Internet bank.

Computerized audit files also create a record of every electronic banking transaction that occurs within VBM. Such log files are used to detect suspicious activities and are used as audit trails to detect and prevent fraudulent transactions. Log files include information such as the customer's name, Internet address, account number, the time and date of each transaction, the payee and the transaction amount. Specifically, the log files and the audit trails they document are designed to detect types of fraudulent activities such as (a) access to multiple accounts from a single network location; (b)

unusually high volume of transactions against an account; (c) unusually high values of transactions; and (d) unusually high amount of bill payments to one payee.

Virtual Vault also provides an authorization mechanism that splits administrative tasks among administrators that have different privileges for different aspects of system administration (e.g., backup, web site configuration). This helps eliminate the possibility of attacks that are facilitated by collusion of an insider, one of the most prominent types of security crimes.

A final level of security exists within the Internet banking application. Here, VBM provides authorization payment limits at both the bank and customer level. When a payee-based limit is exceeded, the VBM's internal audit system generates an alert to a designated individual or group of individuals. The alert details the payee, the limit and the current amount scheduled to be paid, for use by the bank to perform further investigation. If no suspicious activity is detected, the payments are transmitted as normal to the bill payment service. This provides a mechanism whereby the bank can monitor and detect electronic payment activity that is outside of the norm of what is actually performed by the customer. This would be only possible if an intruder somehow stole a customer's password -- potentially by a client side trojan horse, for example. This is something out of the control of the bank's security architecture, but it is monitored for and detected by the application

6 Project Partners

Several organizations collaborated with Hewlett-Packard as project partners to bring everyone's collective expertise together to deliver a successful Internet banking project. From a customer relationship perspective, HP was the primary systems integrator and most of the project partners acted as sub-contractors to HP on the engagement. Each of the partners contributed their value-added expertise as necessary, and as outlined below.

- *HP Financial Services Professional Services Organization (PSO)*
- *Security First Technologies (formerly Five Paces Software)*
- *Bluestone*
- *HP ISSL*
- *Alltel*
- *Vantage One*
- *Client IS Personnel*

7 Recommendations and Conclusions

Based on this Internet banking experience, several recommendations are given on areas of best practice as well as precautions on areas of project risk. Some of these are recommendations that are general in nature, while others are specific to certain types of implementations. Further, some of these are technology related, while others are people and process related.

1. Legacy data access is the most complex task for Internet banking integration projects. It is recommended to choose the mechanism, and design the architecture as soon as possible. This requires a good methodology and customer participation. In our project, we could have compressed the technology interviews.

2. IBM ECI is a nice way to access legacy data without modifying CICS transaction programs. This promotes reusability of customer CICS code. When it is available, it is suggested to use CICS 2.0 with the *DCE-lite* (e.g., RPC only) option for less configuration complexities. IBM seems to be committed to this API, and has web-enabled it. Our client is also very committed to this API.

3. It is important that adequate time is spent with the final release of the Internet banking application software for testing and operational support reasons.

4. It is crucial to fully document Test Plans that are used by the project teams to unit test, regression test, and integration test the Internet banking system.

5. It is equally if not more important to fully document an Operational Support Plan that provides detailed information on managing the Internet bank:

- how the system operates,
- the flow of data through the system,
- data processing flows, schedules, and procedures
- security procedures,
- system logs and error logs,
- and any areas for human intervention.

Further, bank personnel need to be identified to perform these functions, as well as how they will interact and communicate with the rest of the project team. It is important in this Operational Support Plan to document the cross-organizational communication and processes that are required. For example, when an error is reported with the system, a process needs to be initiated by the Help Desk personnel that could potentially involve Operations and System Development personnel to help locate and solve the problem.

6. It is recommended to not implement many new projects and types of integration, simultaneously. The bank assumes much greater risk for the success of the Internet bank project completing on time and on budget, since it is dependent on the success of each of these other projects. Examples of tangential projects that produce a potential dependence include:

- Internet Bank Delivery Channel Project
- Electronic Billpay Project
- Synchronized Billpay
- Corporate Internet connection
- Corporate Email
- Network/Systems/Application Management project

7. It is important to architect and integrate subsystems into the Internet bank architecture in such a way as to not adversely affect the performance, availability, and reliability of the banking experience for customers. For example, extreme caution should be exercised when implementing synchronous communications to external systems, such as billpay providers, into the application. Besides the problem with unavailability, slow response times for the bank customer could occur and lead to the customer not using the bank's Internet delivery channel. An asynchronous communications model for backend communications has the potential for a better customer experience.

8. The development model for an Internet bank differs greatly with the standard waterfall project development model. Specifically, Internet development projects are typically characterized by the concept of *web time* and getting new capabilities to market as quickly as possible:

- Rapid application development and integration
- Phased in functionality and capabilities over time (e.g., evolutionary development) to decrease time to market

One analogy is with the way that Netscape came to dominate the Internet browser market by quickly bringing to market browser technology and phasing in very quickly additional capabilities. Netscape would not have been nearly as successful if it had used the standard waterfall model.

9. The performance of the *overall* Internet banking system should be modeled and simulated to eliminate the bottlenecks as much as possible. Many factors will influence the overall Internet bank system performance, as perceived by the customer. Some of these are:

- Internet delay over 28.8 or 14.4 modem.
- Internet Service Provider delays
- Firewall performance
- Application database performance
- Synchronous communications response times
- Other load on application servers
- Customer usage

10. Methodology is crucial to success in Internet banking projects -- especially in the beginning of these projects -- due to their extremely high visibility inside the bank. An executive workshop elicted project principles, themes, and requirements from key

executive stakeholders. These project principles that are so unifying and global that they guided the development of the remainder of the business requirements. Next, all stakeholders brainstorm on the various business requirements for the project and consensus is gained on which of these are absolutely essential for the various phases of the project, and which of these would be added value functionality. These project principles and functionality requirements are then mapped to the real world by understanding and taking into consideration the existing technology infrastructure of the bank. This leads to certain gaps in the planned system which is documented in a gap analysis, and which needs to be resolved in the final overall system design.

11. When problems cause the project to stall, the project steering committee or project executive sponsor needs to be informed to act as the intermediary so that the problem may be quickly resolved, rather than delaying the project deadlines. This is one of the most important responsibilities of the steering committee and executive sponsor.

12. Distributed CGI processing, that is processing by CGI programs that then make remote calls (e.g., Berkeley sockets, DCE, transaction process monitor) needs to be well coordinated. This is due to several factors. First, concurrency could be introduced on a single CGI and this concurrency of remote processing needs to be synchronized when the HTML is returned back to the browser. Another reason, is that the user interaction with the web is inherently asynchronous (e.g., the user can press Stop and Reload at any time), and these "exceptions" need to get propagated to the remote processes, so that they may be interrupted and possibly terminated.

13. It is important, at project startup time, to clearly define project team roles and responsibilities and to secure team member commitments on their contributions to the project. This is crucial to avoid any ambiguities, overlap, and redundancies.

14. It is beneficial to try to encapsulate legacy data access for optimum reuse of legacy data access throughout this project and other related projects that use the same data. For example, when the VBM Internet banking application changes from a batch model to a real-time access model, if the application is architected such that the data access is encapsulated with a well-defined bank business API, then much of this can be reused and will not have to be rewritten.

15. When a bank goes to a third party for application software, -- in this case Internet banking software -- typically little knowledge or expertise exists within the bank on that application. As such, when issues do arise, it is hard to determine what is critical, what is important, and what is less significant. Further, the relationships and dependencies on the various issues are hard to determine. Finally, little planning can be done to help resolve or work around these issues, without this knowledge. Therefore, having a primary systems integrator or applications consultant dedicated to the project is crucial. The integrator has the application expertise and serves as the single point of contact between the customer and the application vendor, as well as with any other third parties and sub-contractors.

References

1. Hirsch, Bernie: *Accelerating Virtual Banking by Architecting for Alternate Delivery Channels*, HP internal publication, 1996.
2. *Enterprise Solution Model* presentation, HP internal publication, 1996.
3. *Client/Server Programming*, CICS Family: Client/Server Programming, SC33-1435-02, IBM Programming manual, Third Edition, March 1997.
4. *VBM System Integrator's Guide*, Security First Technologies, 1996.