Integrating Financial Data over the Internet

by

Howard W. Pan

Submitted to the Department of Electrical Engineering and Computer Science

in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Electrical Engineering and Computer Science

at the Massachusetts Institute of Technology

May 7, 1999

[June 1999]

Author____
Department of Electrical Engineering and Computer Science
May 7, 1999

Certified by____
Professor Stuart Madnick and Dr. Michael D. Siegel
Co-Thesis Supervisors

Accepted by____
Professor Arthur C. Smith
Chairman, Department Committee on Graduate Thesis

Integrating Financial Data over the Internet
by
Howard W. Pan

Submitted to the
Department of Electrical Engineering and Computer Science

May 7, 1999

In Partial Fulfillment of the Requirements for the Degree of
Master of Engineering in Electrical Engineering and Computer Science

ABSTRACT

This thesis examines the issues and value-added, from both the technical and economic perspective, of solving the information integration problem in the retail banking industry. In addition, we report on an implementation of a prototype for the Universal Banking Application using currently available technologies. We report on some of the issues we discovered and the suggested improvements for future work.

Co-Thesis Supervisor: Professor Stuart Madnick
Title: John Norris Maguire Professor of Information Technology

Co-Thesis Supervisor: Dr. Michael D. Siegel
Title: Principal Research Scientist, Management Science and
Co-Director, Finance Research Center

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Number *Page*

ACKNOWLEDGMENTS

The author wishes to thank Professor Stuart Madnick and Dr. Michael Siegel for their time and guidance on this thesis. They gave me the opportunity to work with a group of very brilliant and insightful individuals developing leading edge technology. I also want thank Min Huang for her contribution to the design and construction of the prototype. Lastly, I want to acknowledge the contributions of Lydia Sandon on the initial draft of this paper, which we began in the fall of 1998.

CHAPTER 1: INTRODUCTION

1.1 The Issue of Implied Data Context in Interpreting Information

Although many Web portals are available today, most, if not all, provide only the most rudimentary level of aggregation and integration of information. Often, Web sites simply collect data from multiple sources and present them, perhaps even in a summarized format, in a central location. This simple view of integration is problematic as information is always expressed within a given context, often embodying a number of unstated assumptions that are well known to members of the same domain. Integrating information, while ignoring its context, can lead to incorrect results when either the integrator or the user is not familiar with the underlying contexts. Worse yet, the user may not even realize an error has occurred.

The widespread usage of the Internet exacerbates the problem as it enables a reader in the U.S. to access information from around the world more easily than ever before. For example, when the online version of the *South China Morning Post*, a Hong Kong-based English newspaper, quotes a company's sales in dollars, the quote is given in terms of the Hong Kong dollar and not the U.S. dollar. At the current exchange rate of 7.75 HKD to 1 USD, ignoring the context can greatly distort the reader's perception of the company's actual sales. Even within a purely domestic environment, there are many domain-specific contexts that may be unfamiliar to outsiders. For example, bond prices are often quoted in the form of "XX.YY". Bond traders know to interpret the quote as XX YY/32 percent of the face value of the bond while the lay person may incorrectly interpret it as XX YY/100 dollars, which is the usual interpretation. As bond trading moves online, more and more novice investors will run the risk of misinterpreting the information and forming wrong conclusions.

1.2 Related Work

Many research projects have studied the problem of contextual information and its relationship to integration. At the Sloan School of Management, for the example, the Context Interchange Group has built several applications to serve as platforms for developing the necessary technologies and a better understanding of the issues of integration specifically in the securities and management consulting industry [MMS98].

1.3 The Objective of the Thesis

The objective of this thesis is to study the emergence of financial portals, Web sites that aggregate financial data, from the perspective of retail banks. We want to understand the social, economic, and technological challenges of integration in this domain.

1.4 The Specific Research Goals

Our goal is to:

1. Identify an interesting problem related to the integration of information
2. Define a specific application that addresses the problem
3. Develop a set of scenarios illustrating how such an application can be used
4. Evaluate, develop, and refine existing technologies for integration
5. Implement a prototype demonstrating the feasibility of our proposal.
6. Outline the business value of such an application

1.5 Contribution of the Thesis

Based on an analysis of the strategic threats faced by retail banks and the problems faced by consumers today, we proposed a Quicken-like application, known as the Universal Banking Application (UBA), to assist individuals in the management of their personal financial information.

Our application improves upon Quicken's functionality in multiple ways. First, we do not require the participating banks to conform to a single standard, providing instead, a flexible system that works with a wide range of content, capabilities, and requirements. Second, we provide intelligent integration of information instead of simple aggregation. This functionality becomes increasingly important as personal finance is integrated into the global commerce. Third, we provide a set of different functionality that does not exist in Quicken.

We have implemented a prototype demonstrating the feasibility of the concept and received significant interest from the retail banking community. Based on the feedback and the lessons learned, we plan to expand upon the work further this summer.

1.6 Structure of the Thesis

In the chapters that follow, we will first present a brief overview of the recent changes in the financial services industry as a consequence of the deregulation of the industry in the 1980s. We outline the problems that the consumers experience today as they face an ever-expanding set of financial products from which to select and describe how the UBA helps to resolve some of the problems. In Chapter 3, we present a conceptual overview of the UBA. In Chapter 4, through a series of scenarios ranging from the simple reading of a single account to the more complex transfer of funds across institutions, we illustrate the value gained through the use of a UBA. In Chapter 5, we present the design and implementation of a prototype of a UBA to demonstrate some of the possibilities. In chapters 6 and 7, we discuss some of the technologies in existence today and how they can assist in the process of integration. In Chapter 8, we analyze the different business models that facilitate the adoption of the UBA. Lastly, we present a list of suggestions for future research.

CHAPTER 2: BACKGROUND OF RECENT CHANGES IN THE FINANCIAL INDUSTRY

2.1 Evolution in the Management of Personal Financial Information

As a result of the intense competition in the aftermath of the deregulation of the banking industry, individual consumers have become increasingly more sophisticated in their choice of financial products. It is typical these days for a person to have a credit card issued by a different institution than his/her provider of checking and savings accounts.

For example, in the figure below, the user is a customer of three different financial institutions. He has a checking and savings account from Bank Boston, uses a Citibank credit card, and invests in the stock market through Fidelity Investments. His company, Widget, Inc., directly deposits his salary on a biweekly basis into his checking account. He then withdraws money and writes checks to pay for various purchases and credit card debts.

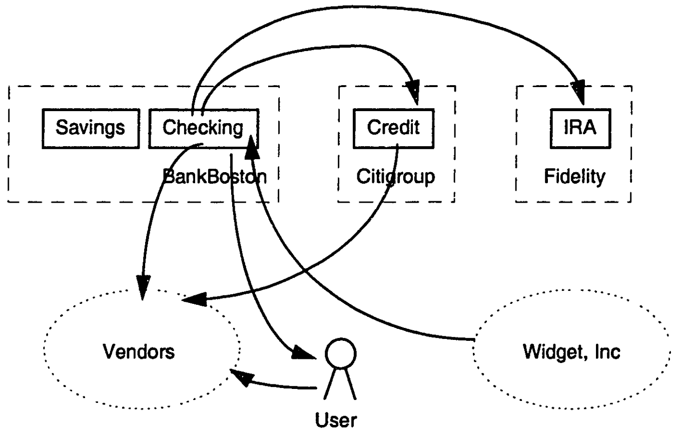

Figure 1 Capital flows for a typical individual

Naturally, the management of personal financial information in this new age is a difficult task. Unfortunately, the individual financial institutions have been slow in providing assistance. John Hagel III, a partner at McKinsey & Company, notes that since a bank's "information system...and organizational structure tend to emphasize products," it is difficult for banks to adopt an integrated focus. They are "hampered by being able to 'see' customers only through their use of [the banks'] products and channels" [HHH97]. As Bowers and Singer point out in their 1996 *McKinsey Quarterly* article, a "widening gap between the information that financial institutions actually provide to their customers and the information that might be delivered" is developing rapidly [BS96].

Figure 2 depicts the various ways our user in Figure 1 may try to manage his financial data today. Since the user uses his credit card and checking account to pay for most of his purchases, these two accounts generate the most transactions. As a result, the user accesses the accounts electronically. He downloads the account data from Bank Boston in to a personal finance management program (e.g. Quicken), and views the credit card transaction over the Web. For less active accounts, such as his savings and brokerage account, the user simply checks the monthly paper statements for errors and files them in the cabinet.

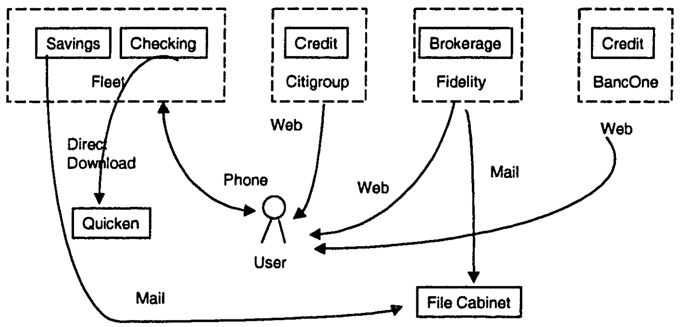

Figure 2 Typical management of an individual's financial data

As a result of the laborious and tedious work necessary to reconcile the accounts, the user maintains minimal control and does not have an accurate nor up-to-date view of his financial status. For example, there is no simple way for the user to determine whether he is operating within pre-defined budgets. Moreover, categorizing his expenses and preparing his tax filings are other examples of such difficult tasks. Most people can appreciate the problem described here especially if they have ever shuddered at the suggestion of spending an evening figuring out their accounts.

The recent shift from providing paper-based statements to using the Internet (or some other PC banking software) provides a valuable opportunity for banks to close this gap. Unfortunately, the problem remains unresolved as each financial institution provides its own interface for accessing information, each of which cannot be linked easily to the other.

2.2 The Bank's Dilemma

Banks are not irrational. They clearly understand the benefits consumers gain through an integrated service. However, there are many reasons for not providing such an interface. In addition to the difficulties involved with creating an industry-wide standard, "most American banks view electronic networks as an opportunity to lock in their customers by creating deeper relationships through more frequent [and] less expensive interactions" [HHH97]. An approach based on open standards eliminates the switching costs they worked so hard to build. As a result, bank sponsored research in this area has focused mainly on means to transform existing strengths of the banking industry into their electronic counterparts. Representative work here includes the Financial Services Technology Consortium's (FSTC) projects on Electronic Checks (eChecks) and Bank Internet Payment Systems (BIPS).

2.3 Impact of New Software Entrants: Quicken and Money

As is the case with most industries, an unmet demand will attract new entrants who are eager to profit from the opportunity. In an attempt to help consumers close the information gap, software companies, such as Microsoft and Intuit, entered the market with products (Money and Quicken) that provide some of the necessary integration service. Again, Hagel warns that these companies are focused on "the acquisition and management of customer [financial information]...and are becoming a new form of intermediary that 'unbundles' the customer [information] management element of [personal financial service] from the product manufacturing and processing elements" [HHH97].

> [Their] brand, customer base, and software expertise give [them] the clout to shape a web of banks, merchants, and technology partners. [Their] web includes leading players in each product segment, including Schwab and Fidelity in mutual funds, [E*Trade] in online trading, CheckFree in electronic payments, and over 50 major banks in credit and transaction products.... By working with many financial institutions and information providers and leveraging their expertise, brands, and products, [they are] able to augment [their] capabilities and deliver seamless personal financial services to customers. [HHH97]

In other words, by providing the highly desirable integration service, the software companies may displace banks of their traditional role as the manager and advisor of an individual's financial needs and transform a bank's products/services into commodity items. To fight back, banks must take immediate action to address their customers' needs. By developing an open standard for financial data integration, they can wrest control of their distribution channel from the new entrants.

CHAPTER 3: PROPOSAL FOR A UNIVERSAL BANKING APPLICATION

We propose in this chapter an alternative architecture, the Universal Banking Application, for conducting financial transactions and managing financial information.

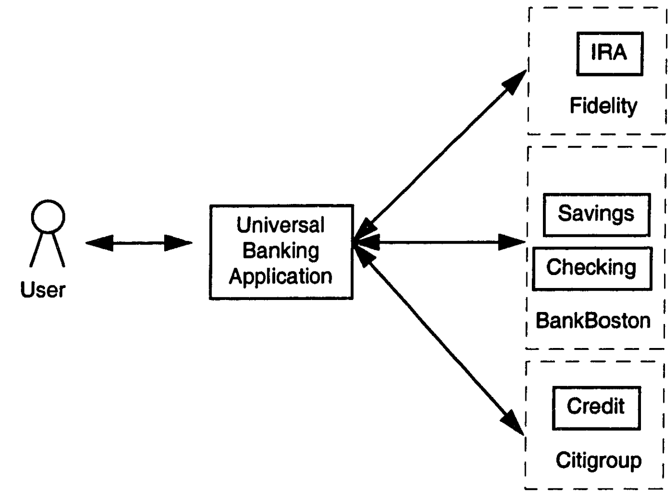

Figure 3 Role of the Universal Banking Application

The UBA can be best described by contrasting it against the *status quo.* Whereas currently, individuals maintain an independent and separate relationship with each financial institution, the UBA presents a unified interface for the customer to all the participating institutions. Figure 3 diagrams the role the UBA plays in mediating the relationship between an individual account holder and the financial institutions. In essence, the UBA becomes the underlying conduit for conducting financial transactions and account inquiries.

3.1 Features of the UBA

There are some immediate and obvious advantages for an individual to use the UBA. On an individual account level, because transactions and inquiries are conducted electronically, it becomes easier to reconcile statements, retrieve information, and make payments. Moreover, when individuals conduct business with multiple institutions, the UBA presents a single interface to the user. Immediately, one sees that it becomes easier for the user to interact with the various institutions. Furthermore, since a unified interface is available, the individual can integrate data from multiple accounts and get an overall view of his financial status. For example, the user can easily identify all tax-deductible spending even though the data is available through different sources. Figure 4 highlights some of the features that should appeal immediately to the average consumer.

Universal Banking Application Feature List

Individual Accounts
- Balance checkbook faster, more accurately
- Send and track payments online
- Pay recurring payments automatically
- Transfer funds

Integrating Data from Multiple Accounts
- Track income and expense easily
- Generate realistic budget plans
- View net worth, budget progress in one place
- Alerts to opportunities such as lower credit card rates
- "What-if" analyses
- Find potential tax deductions, facilitate tax reporting

Third Party Integration
- Export data directly into Excel

Figure 4 Universal Banking Application Feature List

The benefits of the UBA go beyond those listed in Figure 4. More specifically, the adoption of the UBA improves the underlying architecture in many subtle and non-obvious ways. This will be the topic of the next chapter. However, it will be necessary to briefly outline the UBA architecture before we continue.

3.2 UBA Architecture

The UBA, as diagramed in Figure 5, is a multi-tier, client/server architecture made fault-tolerant through redundancy.

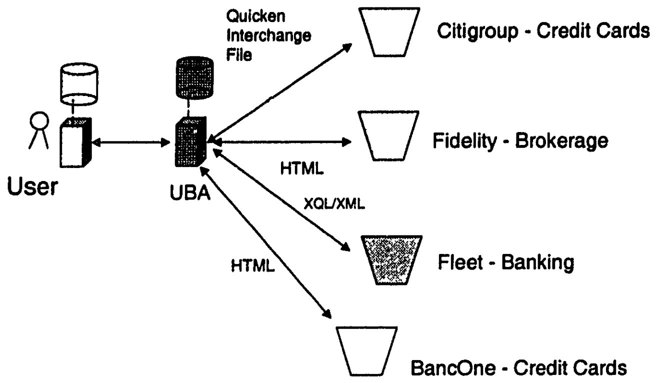

Figure 5 UBA Architecture

The individual financial institutions are allowed to provide interaction and information through any existing format and to any extent of their current capability. For example, some institutions may be providing data through the Quicken Interchange Format. Others provide simply the ability to display data on a Web browser via HTML. Some may provide the entire history of the account while others may provide only information within the last 30 days. The UBA encapsulates the existing services and functionality and can provide through additional storage and computation a unified level of service. Naturally, the user may have additional computational power and storage locally; therefore, the UBA also provides the ability to integrate its services with those located on the user's computer.

3.3 UBA v. Existing Personal Finance Applications

The UBA improves upon the existing functionality of personal finance applications in several important ways.

First, the UBA does not require the participating financial institutions to conform to a single standard. There are probably as many different corporate policies as there are banks in the industry. Instead of providing a single solution that is expected to fulfill everyone's requirements, the UBA provides a flexible system that will work with a wide range of content, capabilities, and business requirements.

In addition, the UBA extends this flexibility to the end users as well. The UBA does not assume that its users have the same needs, agree upon the same definition of terms, or even share the same personal profile. Instead, it provides intelligent integration of information displaying the information according to each user's circumstance. This functionality will become increasingly important as personal finance is integrated into global commerce.

Unlike existing personal finance software, the UBA's design works well in a distributed environment where the participants' differences are accounted for through an understanding of the context of integrated data.

CHAPTER 4: THE VALUE OF INTEGRATION

In this section, we provide a detailed analysis of the features and functionality of the UBA. We begin by analyzing the simplest scenario where the individual only has a single account. We present a list of value-added functionality that the UBA provides over the *status quo.* We then repeat this process with progressively more complex and sophisticated scenarios of working with multiple accounts and conducting transactions between them.

4.1 Reading From a Single Account

Reading from a single account is the simplest and most basic scenario. The user only has one financial account and the only functionality required is account inquiry. For checking, savings, and credit card accounts, users will typically be interested in reading the individual transactions as well as the total account balance. For example, they may wish to get a list of the checks issued, sum of the funds deposited and withdrawn, and total financial charges accrued. For brokerage accounts, the users will also be interested in getting a list of individual buy and sell orders as well as the total account balance. The user may also want to know of their holdings in any particular position, their current margin balance, and the available fund in addition to the overall net worth.

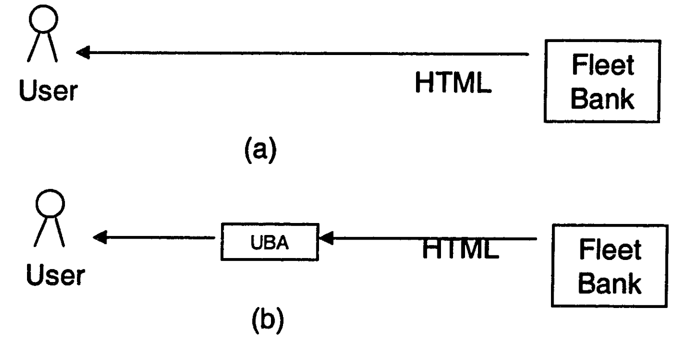

Figure 6 Reading from a single account

Even in this simple scenario, the UBA is able to provide valuable services unavailable today. Specifically, the UBA can enhance the security of the system, provide better timeliness and consistency of data, and improve the availability of the system.

4.1.1 Enhanced Security and Authentication

Security and authentication help to ensure the user's information is not divulged unintentionally to a third party.

Status Quo

Under the *status quo*, security and authentication is arranged independently between the user and the financial institution. This is represented as Option 1 in Figure 7.

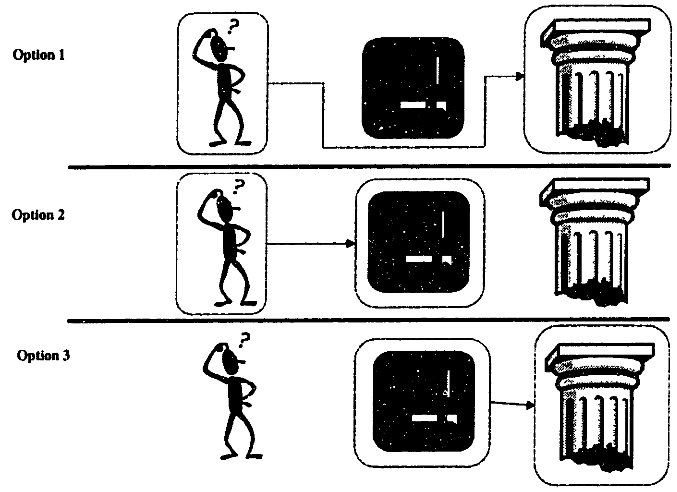

Figure 7 Security Models

The UBA Solution

With the UBA, security and authentication is mediated by the UBA. In general, there are two different approaches. The first approach envisions trust between the UBA and the financial institution. In other words, the UBA presents a unified interface for secure transactions as shown in Option 2 of Figure 7. The second approach envisions trust between the user and the UBA. The UBA becomes the user's agent in handling financial matters as shown in Option 3 of Figure 7.

4.1.2 Enhanced Timeliness of Data Available

Timeliness refers to the ability to access information for any specific time period. For example, a user may want to know the beginning balance in his investment account for the current year.

Status Quo

Under the *status quo,* the user can only access information with the timeliness provided by the individual institution. However, different institutions often provide different levels of timeliness. Some may only provide the current balance, where the definition of "current" differs from institution to institution, and require the user to derive the balance as of any date by deducting interim transactions. Since most institutions provide on the most recent history, calculating the beginning balance may be prohibitively difficult.

The UBA Solution

Since the UBA architecture allows for storage at the UBA server, the UBA can store historical transactions beyond that provided by individual institutions. Given the availability of past transactions, any level of desired timeliness is possible. Moreover, users will be able to expect a consistent level of timeliness of information regardless of the financial institution.

4.1.3 Enhanced Availability of Account Querying Systems

Availability refers to the ability to service the user requests as they arrive.

Status Quo

Under the *status quo*, users interact with different institutions individually. As a result, if the system's availability is limited.

The UBA Solution

Just as with timeliness, storage of past transactions allows the UBA to provide backup service if the services of individual institutions become temporarily unavailable.

4.2 Reading from Multiple Accounts

Clearly, the scenario we painted earlier is too simple. Most individuals maintain a relationship with a multitude of financial institutions. It is not longer uncommon for a person to have a checking account with one bank and use a credit card from a different company. In this section, we consider the benefits gained from adopting the UBA for customers that have multiple accounts.

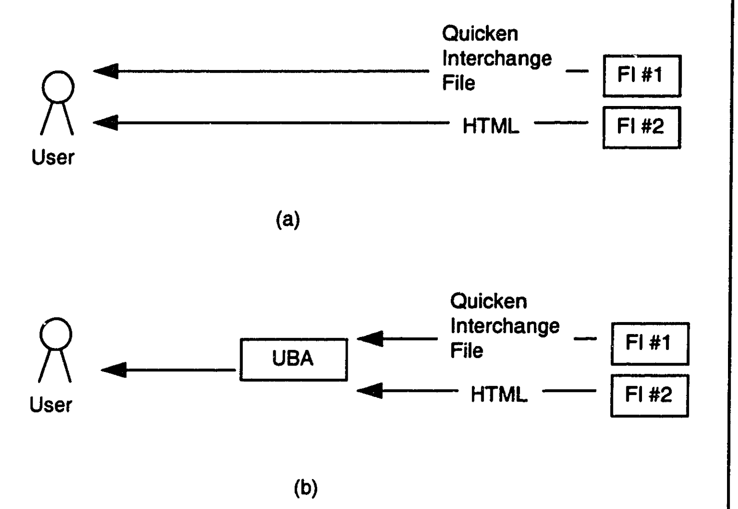

Figure 8 Reading from multiple accounts

4.2.1 Consistency of Account Query Interface

Although institutions provide information and account query capabilities across the Internet, there are vast differences in terms of the range of services. Some data sources are database driven, and thus, can answer complex, user-defined queries while others only provide the ability to provide pre-canned results. Moreover, there is a difference in the amount of data available from each source. Lastly, the data and account query service may use different applications.

Status Quo

Currently, there are vast differences in the capability of, the amount of information available from, and the type of applications used at different financial institutions. As shown in Figure 8 Option A, one financial institution allows the user to download the account data via the Quicken Interchange Format while the other institution provides a Web-based service. As a result, the user needs to install and maintain two different applications on the desktop. Since Quicken allows local storage of historical account data, the user has access to more data than that provided by the data source. Moreover, since Quicken is a more developed product, we assume that Quicken provides more sophisticated querying capabilities. This spectrum of service levels makes it difficult for the users to access the multiple accounts.

The UBA Solution

Through the use of additional storage and computation power at the UBA server, the UBA architecture provides a unified account query interface for the end-user. The interface is unified in two different ways. First, regardless of the capability of the individual data source, the data sources, as seen through the UBA, will always provide the same level of service. The user is guaranteed that the same amount of data, both in terms of detail and time period, is consistent across all accounts. Second, the interface provides a standardized interface for querying different accounts. This reduces the amount of learning necessary.

4.2.2 Consistency of Data Context

Data is always provided with an implicit and unstated context. For example, people often leave off the area code for their telephone number when it is clear to the listener the implied area code. The implication may result from outside of knowledge of the speaker's location. As a result, any program that has access to multiple data sources must be capable of reconciling the semantic conflicts among sources and receivers. This problem is generally referred to as the need for *semantic interoperability* among distributed data sources.

Status Quo

Currently, there is no mechanism for expressing the context to which the data is associated. As a result, the user must understand *a priori* the context for each data source. For example, when viewing the current balance, the user must know beforehand that the "current balance" for some accounts are only valid as of the close of business from the previous day while the current balance for other accounts are valid as of the current time.

The UBA Solution

The UBA can provide two different solutions to the problem of data contexts. The first method envisions that all financial institutions agree upon a single standard for defining all data reported. The second method uses the idea of context mediation.

Context mediation is the process of automatically reformulating user queries posed in the user's context into a set of mediated queries where all potential context conflicts are explicitly solved. A context mediator detects and reconciles semantic conflicts among distributed data sources and receivers. Users are offered transparent access to the remote information sources through a context mediator and gateways to the data sources.

4.2.3 Integrating Multiple Accounts

Although the ability to access multiple accounts through a single interface is useful in its own right, often the information of real interest comes from compiling and summarizing data from several different sources. For example, when faced with a purchasing decision, knowing the total amount of cash on hand is just as important as knowing the current balance in the checking account. Clearly, the ability to provide the user with a more complete financial picture provides added value not available through the simple access of multiple accounts.

Status Quo

Unfortunately, this is a very difficult and laborious task under the *status quo*. The user must first import data from each account individually. If the financial institution's service does not provide an interface for exporting data, the user must key-in the information manually. In addition, the user must also know the separate data contexts for each account and reconcile the differences manually.

The UBA Solution

Through the use of a context mediator and a standard account query interface that hides the differing capabilities of the different financial institutions, the UBA is able to easily provide an integrated view of the user's financial status. Giving the user the integrated view of his financial status is merely a simple extension to existing functionality for reading from multiple accounts.

4.3 Transacting Within One Institution

Thus far, we have only analyzed the value added for read-only functionality. In the next two sections, we expand the analysis to include transactions. Specifically, we will analyze the value added for the transfer of funds between accounts.

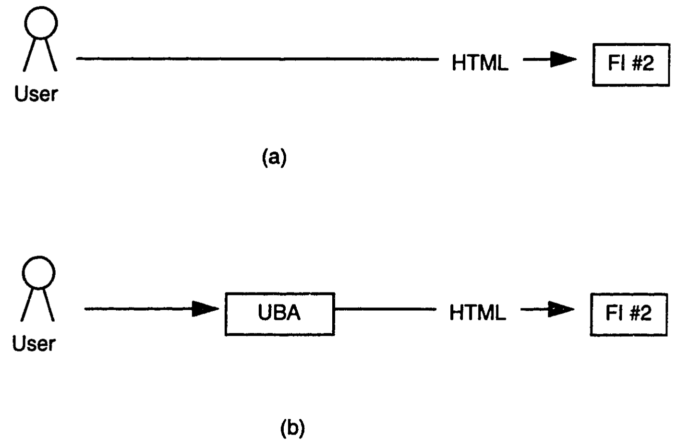

Figure 9 Transasacting within the same institution

We assume that transactions that occur within the same institution execute differently than those between institutions. Specifically, we assume that transactions within the same institution are cheaper, faster, and simpler to setup as compared with transactions across institutions.

For simplicity's sake, we will begin our analysis with users who want to transact between multiple accounts within the same financial institution.

4.3.1 Timely Reporting

Timely reporting refers to the ability to see the results of the transaction quickly. For example, the user should be able to see an increase in his current balance immediately after he deposits funds into his checking account.

Status Quo

Under the *status quo*, different financial institutions update the reported information at different frequencies. For some institutions, the transaction will only be posted on the following day if the transaction takes place after 3PM while others update their reported information continuously. Moreover, even among those institutions that do not provide up-to-date information, there are differences regarding the time at which the information is updated. As a result, timely reflection of the results of the transaction is not possible.

The UBA Solution

Since the UBA has access to the individual accounts and since the transactions pass through the UBA, the UBA can augment the reported information provided by adding transactions that have occurred but not yet posted.

4.3.2 Other Benefits

Naturally, providing timely reporting on the transactions is not the only value added by adopting the UBA. The UBA provides additional value in the areas of providing a unified account query interface and augmented security. However, since we have discussed these benefits in the sections above, we will not go into more detail here.

4.4 Transactions Across Institutions

In this section, we expand the functionality to include transactions between different institutions. The UBA provides a variety of added benefits such as a reliable transport network, cost minimization, and the ability to check on the status of the transaction.

4.4.1 Reliable Transport Network

A reliable transport network allows the user to be certain that if funds were deducted from an account, the same amount has been deposited into the corresponding account of the transaction.

Status Quo

Currently, there are few transport networks that are reliable. The user is required to manually verify the originating and receiving institutions received and executed the transaction correctly.

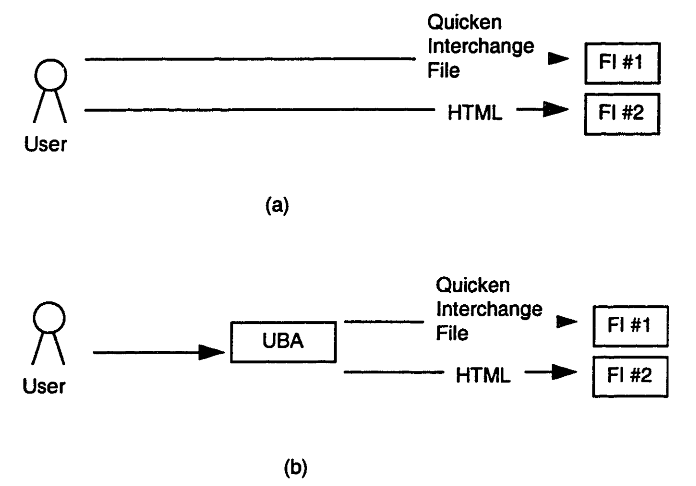

Figure 10 Transacting between different institutions

The UBA Solution

There are two possible scenarios under the UBA architecture. Under the first scenario, the UBA uses existing transport networks. The UBA simply automates the user's tasks and verifies that the originating and receiving institutions received and executed the transactions correctly. Under the second scenario, the UBA provides its own transport network. This allows the UBA to eliminate any errors induced by an intermediary institution.

4.4.2 Minimizing the Cost of Transaction

In addition to selecting the appropriate level of service, users also want to minimize the cost given the level of service desired.

Status Quo

Under the *status quo*, the user decides on which method of transaction to employ. As a result, the user is burdened with the need to know and understand the different costs and level of services for each type of transaction. Not only is this inconvenient and burdensome but it also leads to potential inefficiencies from imperfect information.

The UBA Solution

Since the user only interacts with the UBA, the UBA can help the user minimize transaction costs by selecting the optimal method for transaction. In addition, new methods for transaction can be added without any impact on the users.

4.4.3 Verification of Transaction Status

Since most transactions do not execute instantaneously, users will find it useful to be able to check on the status of any transaction that has not yet completed. This is analogous to the service Federal Express provides on its Web site that allows its customers to check on the status of any package.

Status Quo

Under the *status quo*, getting the status on any particular transaction is difficult. Although individual institutions may be able to acknowledge receipt of payment, there is no information on the transaction as the payment is in transit.

The UBA Solution

Under the UBA, there are two possible scenarios to addressing this functionality requirement. First, the UBA simply uses existing methods of transaction and can only inform the user when the transaction is received by any particular institution. In the second scenario, the UBA provides and uses its own method of transaction. Since the UBA now also serves as the transport network, the UBA is able to provide full tracking of the transaction for the user.

4.5 Integration with Other Systems

Since it is difficult to foresee all of the possible feature requirements, it is important to design the application with interoperability in mind. It is important to provide the user with the ability to integrate the system with other programs.

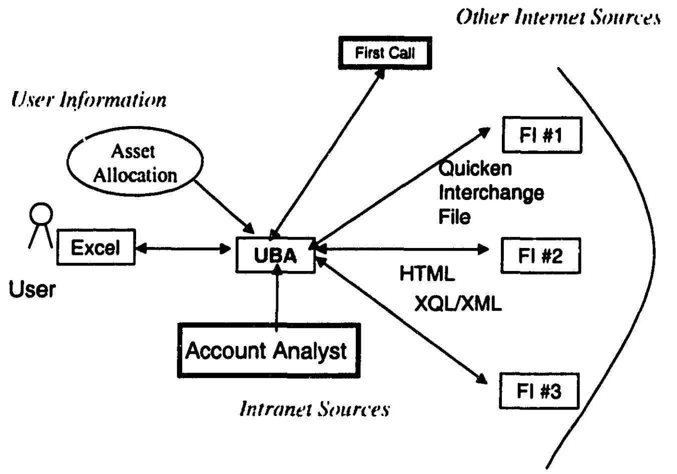

Figure 11 Integration with an outside source of information

4.5.1 *Data Interoperability*

Data interoperability means that the end user should be able to incorporate the output in third party program such as Excel. Users can use the third party programs to enhance the capability of the tool's analytical capabilities.

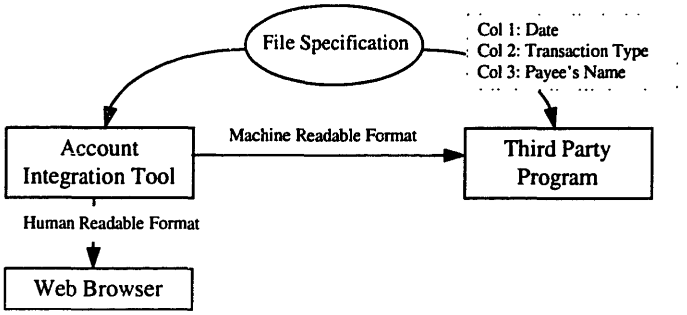

Figure 12 Excel parses "comma separated values" data stream format using the pre-defined file specification.

Status Quo

Under the *status quo*, different institutions may adopt different standards for exporting data. As a result, the user may need to manually convert the exported data from one format into another before it can be read.

The UBA Solution

Since users can access all the accounts through the UBA, users will only need to ensure their application can import the files the UBA exports. This allows the user to achieve data interoperability regardless of the data formats used by individual services.

4.5.2 Programmatic Interoperability

In addition to simply providing data interoperability, the architecture should also be open and accessible programmatically through other applications.

Status Quo

Under the *status quo*, the only way to achieve programmatic interoperability is to provide "wrappers", programs that hide the complexity of the data source behind a uniform interface, for each type of data source.

The UBA Solution

Since the UBA is the only interface the user accesses, a single "wrapper" is needed. The wrapper can be implemented in many different ways including as an object that abstracts the services of the UBA (Figure 13) or as an abstracted set of relational tables.

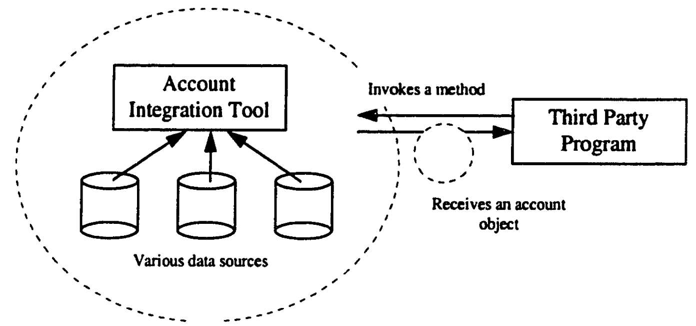

Figure 13 Excel invokes a remote method on the account integration tool object and receives an account object in return.

4.6 Summary of Benefits

The following table summarizes the value provided through the UBA:

Capability	Value Added
Reading from a single account	• Enhanced security and authentication • Enhanced timeliness of data available • Enhanced availability of account querying system
Reading from multiple accounts	• Consistency of account interfaces • Consistency of data context • Integration of multiple accounts
Transacting within an organization	• Timely reporting
Transacting across institutions	• Reliable transport network • Reduction in cost of transaction • Verification of transaction status
Integrating outside data sources	• Data interoperability • Programmatic interoperability

Table 1 Summary of Value Provided Through the Universal Banking Application

CHAPTER 5: TECHNOLOGIES FOR INTEGRATING INFORMATION

In this chapter, we highlight a few of the key technologies that were examined and used in the implementation of the UBA prototype. Specifically, we looked at how wrapper technologies, XML, and XQL can play a role in integrating information.

5.1 Wrapper Technology

A wrapper is a gateway program that transforms the capabilities and interfaces of one set of services into that of another. Wrappers are frequently used in the construction of large-scale computer systems as a way of managing complexity. RPC (Remote Procedure Calls) that abstract the low-level implementation details of network communication away from the high-level task of building client/server systems and object brokers that provide an interface from one system to another are all examples of wrappers used in real life.

Information sources on the Web vary greatly in their interface technology, representation of physical data, and querying capabilities. Some information sources may provide a relational database interface while others perform pattern-matching searches. In addition, many Web sites return their results in an HTML page while others will output the data in a comma-separated, fixed-width, or even XML-based format. For those sites that return information encoded in HTML, some structure the results as an HTML table, while others limit the interface to returning a paragraph like responses. The conundrum of varying interfaces, querying capabilities and data formats suggests that wrappers can be used as an efficient means to abstract away the implementation details of each source and present its client with a unified access interface.

5.1.1 Grenouille and Bon Homme: Regular Expression Web Wrappers from COIN

Grenouille and Bon Homme are the names of two regular-expression Web wrappers developed at the Context Interchange Group at MIT [Qu96]. Both present a relational database-like interface to their clients. Requests are presented in the standard Structured Query Language (SQL) syntax and results are returned in the form of standard objects such as a comma-delimited data stream.

These Web wrappers use a high level declarative language to specify the information content and querying capabilities of each Web-based information source. For each user request, the Web Wrappers' engines use the collection of specifications to transform the request into an execution plan and uses regular-expressions to extract the appropriate data.

5.1.2 W4F (World-Wide Web Wrapper Factory)

W4F (World Wide Web Wrapper Factory) is a toolkit developed at the University of Pennsylvania for generating source-specific Web wrappers in Java that conform to the Domain Object Model (DOM) [SAB99] [SAW99] [SAL99].

There are several important differences between the approach taken by W4F and that of the COIN Web wrappers. Whereas the goal of the COIN wrappers is to build a database engine for the Web, W4F wants to build a flexible HTML to XML document converter. Consequently, W4F objects do not contain any query optimization, query planning, or query execution logic. Although the source accessed can return data that represents the result of a query execution, the W4F objects are only capable of retrieving and interpreting data from that single source. Queries that span multiple data sources must be completed outside of W4F. Likewise, the amount of information, roughly equivalent to the number of columns returned, is fixed for all queries for a given object.

What the W4F lacks in querying capabilities, it more than makes up with a much more powerful extraction capability. The use of regular-expression limits the ability of the COIN Web wrappers from extracting data from complex HTML documents. For example, the nesting of tags creates problems for the COIN Web wrappers, because

HTML documents should be viewed as labeled graphs in accordance with the DOM model [W3C98] and not a simple stream of text. Each document should be parsed into an abstract tree corresponding to the HTML hierarchy. Each node in the tree either corresponds to an HTML tag or text strings. Nodes can be denoted either according to its position in the document's hierarchy or via document flow. This implies that HTML nodes that share the same parent can occur in any order and still represent the same information. It is difficult to build such flexibility into pattern matcher based wrappers.

On the other hand, W4F employs a DOM-conformant view of Web pages. Users can use HEL (HTML Extraction Language) to specify any node in the HTML tree. For example, `html.head.title` specifies the title element in the header section of the HTML document whereas `html->pcdata[1]` specifies the second text string in the document. Moreover, HEL supports dynamic binding of index variables, which allows for a much more robust specification of the information source. For example, the following HEL rule demonstrates the power of the HEL language and the powerful extraction capabilities of W4F.

The rule will:

- Find the table in the Web page that says "Best Movies"
- Find the column index for the column labeled Title and Rank
- Retrieve the title from the Title column for the movie whose rank is 5.

```
info = html->table[i].tr[row].td[title_col}.txt
WHERE html->table[i].tr[0].txt = "Best Movies"
AND html->table[i].tr[0].td[title_col}.txt = "Title"
AND html->table[i].tr[0].td[rank_col}.txt = "Rank"
AND html->table[i].tr[row].td[rank_col}.txt = "5"
```

The design of the W4F objects allows them to be easily coupled with the up-and-coming implementations of XML Query Language. When used together, the two technologies complement one another well and may potentially exceed the performance and capabilities of the existing COIN Web wrappers.

5.2 XML and XML Query Languages

5.2.1 XML

A markup language is a mechanism to identify structures in a document through the use of tags. Markups are commonly used to associate strings with their presentation. For example, HTML allows strings to be enclosed between the <B> </B> tags as way to denote bold facing. The Extensible Markup Language (XML) [BPS98] applies the same technique to the domain of associating data with its semantics. By enclosing parts of the documents between descriptive tags, the XML specification defines a standard way to add structured information to documents through the use of tags and enables programs to extract data more easily.

The value of XML can be best appreciated through a comparison with HTML, one of the most popular and common data formats on the Internet today. Unlike XML, which allows a user to create any arbitrary tag for describing a specific piece of information, HTML comes with a limited set of tags and does not provide arbitrary structure. In addition, HTML associate tags with display and not with the meaning of a particular piece of information. This makes it extremely difficult for a program, such as the Web Wrapper, to identify interesting pieces of information. In fact, early implementations of the Web Wrapper used regular expression-type pattern matching to locate the appropriate data in an HTML page. As we have discussed earlier, this method is fragile as it depends on the physical structure of the document and not on its contents. Even a DOM-like approach that W4F employs is fragile, as the tree traversal is dependent on the presentation of the information that may not be strongly related to the content. In other words, there is no guarantee the maintainer of the data source will not change his mind and present the information using a different set of tags. In comparison, an XML-based implementation of the Web Wrapper would allow a much more robust implementation based on the matching of semantically related tags.

5.2.2 XML Query Languages

Given the immense popularity of HTML as a format for publishing content in a distributed environment, many have similar high hopes for XML, especially in the area of Electronic Data Interchange (EDI). However, XML is not a panacea. The availability of large amounts of XML data will expose several new challenges that XML currently does not address. For example, if XML were to transform the Web into a giant, distributed database, how should a program query this database? XML does not specify a universal extraction language. Moreover, XML does not specify a method for integrating data from multiple sources or for exchanging information between different but related ontology.

The recent XML Query Language Workshop (QL'98), held at MIT, brought together close to a hundred researchers who are studying this very problem [Rein99]. There is a great deal of interest to develop an XML-based query language in order to enable the type of services that SQL provides for databases. An XML query language could make it easier to process large collections of XML documents and extract relevant information.

Although there are various proposals at the workshop, as of this writing, a complete implementation of XML-QL is still not available. When one becomes available in the near future, we would like to further examine the possibility of combining the extraction capabilities of W4F with the querying capabilities of XML-QL.

5.3 OFX

Open Financial Exchange (OFX) is an XML-based industry standard for the electronic exchange of financial data between financial institutions and consumers. Created by CheckFree, Intuit and Microsoft in early 1997, OFX supports a range of financial activities including banking, bill payment and presentment, and investments, including stocks, bonds and mutual funds.

Bank of America, Chase Manhattan Bank, Citibank, Fidelity Investments, and Charles Schwab are a few of the well-known financial institutions that are currently lined up behind OFX.

CHAPTER 6: DESIGN AND IMPLEMENTATION OF THE UBA PROTOTYPE

6.1 Design of the Prototype

Although the UBA can be implemented in many different ways, we chose to implement the prototype as a three-tier architecture consisting of a Web browser running on the end-user's desktop, a CGI-based application running on the Web server, and separate data sources accessible over the Internet. Our goal was to take advantage of the Web-based infrastructure for rapid development. This architecture is depicted in Figure 14. It is important to realize, however, that the functionality of the UBA can reside in any one of the three tiers and is not limited to the implementation described here. In other words, one can just as well have an architecture where the UBA functionality is implemented on the client side. In the sections below, we describe some additional considerations during the design of the prototype.

6.1.1 Variety of Account Types

The prototype currently uses data from five different accounts, one checking account from Cambridge Trust Company, two credit card accounts from American Express and First USA, and one brokerage account from Ameritrade. In addition, the prototype also uses

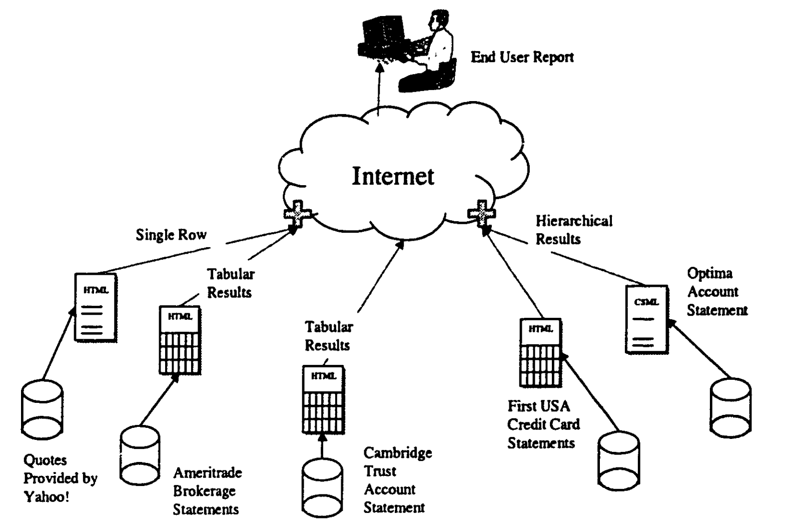

Figure 14 UBA Prototype

fifteen minute-delayed stock price information from Yahoo!.

6.1.2 Variety of Account Information Formats

To demonstrate our ability to integrate data stored in heterogeneous formats, we selected sources that presented its information in a variety of formats. Specifically, we covered sites that are restricted to returning HTML tables and XML-based data. For simplicity's sake, we decided to create our own XML-based markup languages, CSML (Credit Card Statement Markup Language) and myCSML (a simplified version of CSML) for encoding credit card statements, BSML (Brokerage Statement Markup Language) for encoding brokerage statements, and QML (Quotation Markup Language) for encoding stock quotes. The use of CSML and BSML instead of the industry standard, OFX, does not materially affect the design of the prototype.

6.1.3 Typical Access

The prototype currently demonstrates the capabilities of the UBA in the read-only scenarios described in Chapter 4. Specifically, we demonstrate the following:

1. Naïve implementation of simply passing the data source's information as-is directly to the user.
2. Reading from a single data source augmenting the data with that supplemented from another source.
3. Reading multiple data sources, each encoded in its own specific format, and presenting an integrated view of the information retrieved.

6.2 Specific Integration Scenarios

In this section, we present two scenarios that demonstrate the capabilities of the UBA.

6.2.1 Presenting Up-to-date Brokerage Account Statements

As more and more people invest online, the reliability and performance of brokerage Web sites have come under close scrutiny. The media has reported on various system outages at firms such as Charles Schwab, E*Trade, and Ameritrade during times of heavy trading and peak usage. Gomez Advisors, an industry consulting firm in Concord, Massachusetts, estimates that the failure rate of different brokerage sites can be as high as 5%, or roughly thirty minutes per day.

Unavailable systems can lead to severe disruptions for the investor. At best, the customer must put up with annoying delays, and at worst, he/she risks making decisions based on poor data. The author of a recent *Barron's* survey writes:

> Quite a few of our respondents expressed annoyance at the slowdowns and interruptions in service that they have been forced to endure over the past few months of hectic market activity...."What irritates me is the online trading firms' inability to handle current customers while at the same time incessantly advertising for more customers." To protect themselves, [investors] are spreading their bets. Approximately 40% of the respondents said they had accounts with more than one online broker.... Most had two accounts, and a few had three. One reader had five, which strikes us as an administrative nightmare. [Carey99]

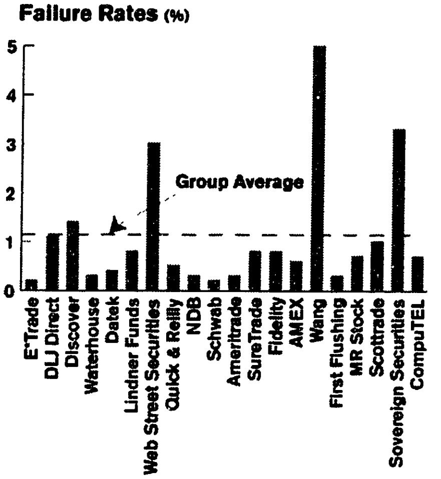

Figure 15 Estimated failure rates of online brokerage Web sites based on Gomez estimates. [Gomez99]

The UBA can provide a valuable service to its users by providing more timely information than what brokerage firms are currently able to provide with their system. By using cached data and ensuring the prices listed in the brokerage statements are consistent with up-to-date prices retrieved from an alternative source, the UBA can provide a more accurate brokerage account statement than what is currently available.

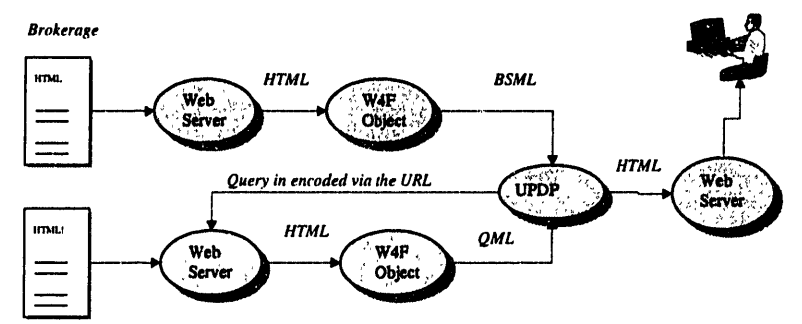

Figure 16 Information and Process Flow for Brokerage Statement Presentation

In Figure 16, we have outlined the information and process flow of the module. The system first retrieves the brokerage statement from the online broker's Web site and parses the HTML document into BSML with the aid of a W4F object. Based on the list of securities in the statement, the module (UPDP) then sends an URL-encoded query to Yahoo! to check on the latest stock prices. The augmented statement with prices supplied from Yahoo! is then sent back to the end-user.

6.2.2 Integration of Credit Card Statements

Students and entrepreneurs often use their credit cards to help meet payment needs. The credit cards offer an easy line of credit, which if the debts are paid off after the first statement comes due, does not carry any cost of interest. As a result, it is often useful and valuable to know the amount one must have in one's savings account to ensure that one will have sufficient cash to pay off this month's credit card debts as they come due.

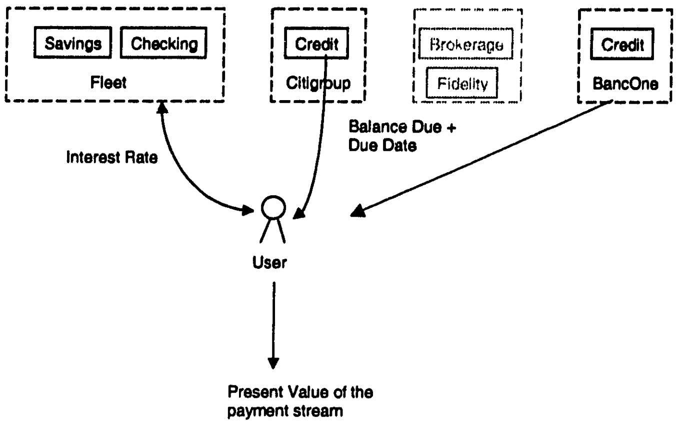

Figure 17 Process of determining whether one has sufficient cash in one's savings account to pay off the credit card debts as they come due.

To accomplish this goal, the user must acquire the balance due and the due date for each credit card account and then discount this stream of payments with the interest rate offered through the savings account. Not only are the calculations not trivial, this is not a simple task, as paper-based credit card statements often do not reflect the latest charges. Accurate and timely account balances must be retrieved electronically over the Web or by placing a call to the credit card company. For the problems mentioned before, integrating this information becomes a tedious and labor intensive task. The UBA can provide a valuable service by presenting an integrated view.

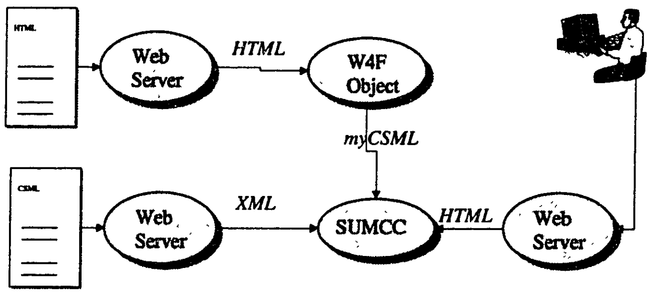

Figure 18 Information and process flow of integrating credit card summary data.

As mentioned before and shown in Figure 18, we demonstrate the ability of the UBA to integrate information that comes from multiple sources each using its own format. The First USA account presents its information in HTML while the American Express Optima account presents its information in CSML, an OFX-like XML-based markup language. Through the use of the W4F Web wrapper, we can automatically convert the HTML data source into an XML like output which our credit card summation module, SUMCC, can interpret.

CHAPTER 7: INTEGRATING INFORMATION IN THE WORLD OF XML

As we mentioned in Chapter 5, XML has been introduced as a method for encoding information so that a computer program can understand and manipulate information without the intervention or aid of a human. Recently, proponents of XML have also suggested that XML can serve as a foundation for exchanging and integrating information as we have demonstrated through the UBA prototype. However, integrating information is not a simple task, and XML does not provide the full solution. As the XML standard evolves, it is important for us to provide feedback and proposals to help resolve the remaining issues in integration.

In this chapter, we focus on the role XML plays in integrating information. In Section 7.1, we analyze the problem of integration and partition it into two disjoint sets, namely that of syntactic representation and semantic representation. XML addresses the problem of syntactic representation but leaves the problem of semantic representation unresolved. In Section 7.2, we review some of the methodologies for resolving semantic representation. In Section 7.3 and 6.4, we analyze the advantages and disadvantages of the different approaches. Lastly, in Section 7.5, we present a simple extension to XML that will address a subset of the semantic problems described.

7.1 Taxonomy of Integration Problems

The domain of integration problems can be broadly partitioned into two different sets, syntactic and semantic.

7.1.1 Syntactic

First, there is the problem of representing the syntax of the information. In other words, before information can be communicated, we first need to have the grammar rules and vocabulary for constructing the structure of the information.

While HTML focuses on the presentation of the data, XML focuses on the communication of the information. Most Web sites today use animated banners, background music, and streaming video to woo viewers, but these additions do not help computer programs to understand the content offered. A person's social security number cannot be differentiated from a telephone number. XML provides a consistent form and methodology for defining the meaning of Web data. In its most basic form, XML enables people to create new tags for describing each data item; thus allowing a program to understand the text enclosed between the pair of <TEL> </TEL> tags represent a person's telephone number.

7.1.2 Semantic

However, understanding the syntax of a document does not guarantee an understanding of the semantics of the information conveyed. A person can understand the grammatical structure of a sentence without understanding its meaning. More importantly, a person can understand the meaning of each word in the sentence but miss the connotations behind each word.

In other words, this is a problem with the knowledge representation aspect of the issue.

However, even within this particular set of problems, there are two different classes of issues, representation conversion issues and reasoning issues.

Representation conversion is the most basic form of semantic problems. Data is often expressed within a given context. For example, in the United States, people measure the temperature in Fahrenheit, and radio broadcasters will often give out the temperature forecast as 70 degrees without specifying the actual units. Binding the context as part of the process of semantic resolution is the first step towards integrating the information.

Reasoning issues relate to the ability to infer additional knowledge based upon the given information. The research in artificial intelligence to date shows that inference capability depends greatly on the underlying knowledge of representation. Although this is an important topic, we will not delve into the issue here. For the remainder of this essay, we will focus on the problem of representation conversion.

7.2 Means Of Resolving Representation Contexts

For the example given above, the context can be resolved in several different ways.

If the listener is from the same domain as the speaker, he will automatically know the units assumed. Of course, this begs the question of how the listener knows whether he is from the same domain as the speaker, especially given that "no one knows you're a dog on the Internet." This approach is often known as *standardization.* We will show below that standardization may not be appropriate in all industries and highlight the criteria of the industries for which standardization may not work.

When the listener is not from the same domain, he can use simple heuristics to determine the context. In this example, the listener knows that 70 degrees Celsius or 70 degrees Kelvin are simply not within the usual range of temperature for describing the weather in Boston. Through the process of elimination, the listener arrives at the proper context. This approach is known as *heuristic inference.*

The listener can also attempt to determine the appropriate context by *assuming* the context for the speaker's domain. For example, given that the broadcast comes from a Boston-based radio station, and Boston is located in the United States, then the proper units for interpreting the temperature reading is probably the Fahrenheit scale as that is the default scale used in the U.S. This approach of resolving the ambiguity by is similar in nature and spirit to how domain names are resolved on the Internet.

Lastly, the listener can always *ask* the speaker to clarify the units used whenever ˙ndeterminacy arises.

7.3 Why Standardization May Not Be A Solution

Most discussions of XML propose to address the semantic problems of integration through standardization. "Industry wide uses of XML ... require a framework for designing multi-vendor documents, one that uses a standard set of message types (like invoice, for-sale notice, purchase order, and catalog query) and a standard dictionary of items used in those messages (like customer name and phone number)" [Zeichick99].

We believe such suppositions are overly simplistic. In a world where there are many different organizations, each with their own policy, set of requirements, and capability, it will be difficult to design a single solution that encompasses the needs of everyone. Moreover, the mapping of organization-specific information to the single standard still requires semantic reasoning. At most, standardization simplifies the size of the problem from $O(n^2)$ to $O(n)$ but does not completely resolve the problem. Lastly, the proponents ignore the past failures in standardization.

The history of technology development is littered with ill-fated attempts at standardization. The failure to create a unified Unix operating system, the numerous browser incompatibilities between Internet Explorer and Netscape's Navigator, and Sun Microsystems' recent law suit against Microsoft for injecting incompatibilities into the Java programming language are all clear proofs that show the difficulty in creating and maintaining a standard.

These attempts at standardization did not fail due to a lack of support and demand from the market. In general, consumers welcome standardization as it reduces the risk of purchase and possibly enhances the value of their product. Standards accelerate the acceptance of new technologies by reducing the likelihood of consumers becoming locked-in to a particular vendor. Moreover, standards enhance compatibility and interoperability, making products with network externality more valuable as the size of the user population increases.

However, there are also strong incentives for suppliers to differentiate their products through the addition of proprietary extensions. Incompatibility creates a switching cost for users and a barrier-to-entry for newcomers. In a winner-takes-all type of competition as is characteristic of the software industry, companies have every incentive to promote their own format, instead of joining together in developing an open standard.

Take AutoCAD for an example.

> According to the Federal Trade Commission, AutoCAD controls 70 percent of the computer-aided design (CAD) market. AutoCAD has used its proprietary DWG format to store files, which other CAD programs are subsequently unable to read and write without translation errors. This is a real problem for architecture firms that need to exchange files with contractors and engineers relying on different CAD applications. [Loizos98]

As a result, the incompatibility problems dissuade existing users of AutoCAD from switching to a competitor's product, and AutoCAD has no incentive in participating in switching to an open standard.

Second, even if a common standard was agreed upon, open standards are prone to fragmentation and hijacking as companies seek to extend the standard with proprietary additions in hopes of gaining eventual control over the installed-base. For example, Microsoft has been accused of trying to extend both Java and HTML in proprietary directions. Economist, Hal Varian and Carl Shapiro, "doubt that Microsoft had much interest in seeing...a unified Java standard...since [the technology] poses far more of a threat to Microsoft than an opportunity" [VS99]. They warn standard setters to "beware of [participants], formally or informally, that deep down have no interest in seeing a successful standard emerge" [VS99].

Therefore, even disregarding the difficulties in assessing the technical tradeoffs and agreeing upon an industry-wide standard, the very nature of competition in a fiercely competitive industry makes standardization an unlikely solution.

7.4 XML Namespace and XML Queries

In addition to the economic an historical argument given in the previous section, there are other reasons to believe that standardization is difficult.

The proponents of the XML namespace "envision applications of...XML...where a document contains markup defined in multiple schemas" [BHL98]. Since it is unlikely that one particular schema will satisfy the needs of all participants within the industry, there will likely be many similar schemas within the same industry. However, since good schemas are difficult to write, and modular design is a good idea, it is useful to re-use parts from existing, well-designed schemas.

Moreover, many proposed designs on a query language for XML have stressed the desirability to yield XML output. "Derived databases (views) can be defined via a single [query]. Query composition and decomposition is aided. It is transparent to applications whether they are looking at base data or a query result" [Maier]. Consequently, it will be very difficult to define a set of industry-wide schema for all possible query-generated result sets.

7.5 Proposal for a Heuristic and Inference-based Context Resolution System

Given that efforts at standardization may fail, we have three alternatives for binding representation contexts -- through heuristic inferences, by assuming the domain's source, and by simply asking the source. We propose a design for binding representation contexts that incorporates the three alternatives.

First, we propose to add a set of sub-elements to the existing tag sets for specifying the contextual information. In other words, any parser of an XML document can always

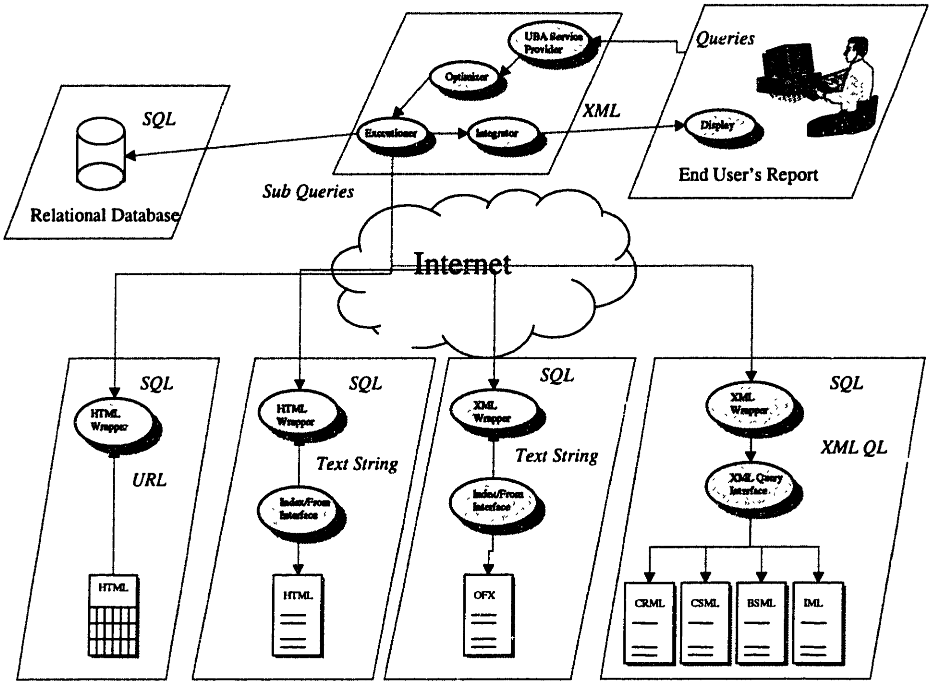

Figure 19 Integration Information in a World of XML

lookup the domain's context if necessary by examining the sub-tags of the given element.

For the example we used above, we could represent the binding as follows:

```
<TEMPERATURE> 70
     <CONTEXT>
          <UNITS> Fahrenheit </UNITS>
     </CONTEXT>
</TEMPERATURE>
```

This approach does not require any extension to the existing XML standard and will assist in the resolution of representation contexts. The tradeoff comes in the increase of bandwidth use.

One possible extension to this idea, though not yet explored, is to extend the DTD instead of the XML document. By loading the DTD, the parser can find the context in one central location. Moreover, it further encourages re-use. Unfortunately, this method does not extend to XML documents that do not have a DTD.

Other permutations of this idea are possible but not explored further.

But even with the best intention, it is difficult to specify the entire domain completely. Therefore, we want to adopt a hierarchical approach to resolving representation contexts. Whenever the context cannot be resolved at the tag level, the resolution process is pushed upwards to the outer level tags.

We extend this idea of pushing the resolution upward beyond the simple hierarchy of tag sets. When the document itself does not contain sufficient contextual information, we may be able to resolve the problem at the level of the author or even at the level of the organization.

CHAPTER 8: BUSINESS MODELS OF INFORMATION INTEGRATION

Thus far, we have only described the technical merits of and the customer's incentives for using the UBA. In this chapter, we focus on the incentives and strategies from a provider's perspective. In Section 8.1, we first present a simple economic framework for evaluating business decisions and understanding the effects of standardization and platform ownership. In Section 8.2, we draw analogies from the personal computer industry and present anecdotal evidence to substantiate the conclusions we drew from our theoretical framework. Lastly, in Section 8.3, we apply the framework to the retail banking industry and present a possible business model for the provider of a UBA.

8.1 Valuing a Bank's Business

One of the most valuable assets that a bank has is its installed base of customers. It represents a steady stream of recurring income in the form of monthly bank charges and interest payment receipts. Often, in bank mergers, the acquiring bank must weigh the present value of the expected income from each customer against the cost of acquiring that customer. For the merger to be profitable, the acquiring bank must be able to derive a higher stream of income than the selling bank. More importantly, the acquirer must compensate the seller sufficiently to compensate the seller's loss of expected income from holding on to its assets. Therefore, by enabling a bank to increase the per customer revenue derived, a bank can increase the value of its business.

Fundamentally, the present value of the income stream is directly tied to the customers' switching cost. The present value of the "profits a supplier can expect to earn from a customer [is] equal to the total switching costs...*plus* the value of the other competitive advantages the supplier enjoys by virtue of having a superior product" [VS99].

In other words, the present value of the customer's cost of doing business with a bank, which equals the profits the bank derives from that customer, must be less than the present value of the cost of doing business with an alternative bank. If this was not true, the customer could improve his/her welfare by banking elsewhere. Thus, any method that increases a customer's switching costs will enable the bank to derive a higher revenue stream.

There are two components to measuring switching costs. First of all, there is the portion derived from existing differences between the products and services offered. Banks can always offer a better product more cost effectively. As mentioned previously, the UBA enables banks to deliver more timely and personalized services than currently feasible. Secondly, there is the portion due to switching costs incurred from switching from one vendor to another. Examples of such costs range from tangible costs such as retraining and reconfiguration of software and hardware systems to intangible costs such as the loss of long running relationships.

8.2 Effects of Open Standards

Industry standards, regardless of open or closed, tend to reduce the customers' switching costs and, in general, lead to a reduction in the value of the bank's business. We say "in general", because, as we shall show later, ownership of an industry standard can greatly improve a firm's competitive position. In the mean time, as a player in the standards game who has agreed to an industry standard, banks have laid down an invariant that the customer knows will remain constant from one bank to another. In essence, it disallows differentiation along that dimension and increases price competition. For example, the fact that most banks, regardless of size, are members of the FDIC (Federal Deposit Insurance Corporation) eliminates one area of possible differentiation, namely deposit safety.

The PC industry provides one of the clearest results of an open standard-based competition. We can broadly divide the industry into two different categories of companies, owner of standards, such as Intel and Microsoft, and users of standards, such Compaq and Dell. All of these are well-known leaders of the industry, yet their performance (as measured by their profit margins) differs widely. Table 2 below shows the significant differences between the performances of the various companies. Because Dell, Compaq, and Gateway all sell Intel-based computer systems that run Microsoft's operating system, there is very little to differentiate one company's products from another. Consequently, their margins are only 20-25% of that of Intel and Microsoft.

5 Year Averages	Intel	Microsoft	Dell	Compaq	Gateway
Gross Margins (%)	65.4	94.4	21.6	26.5	18.0
Pre-tax Margins (%)	38.1	43.7	8.4	4.4	5.9

Table 2 Profit Margins of Various Companies (Source: Microsoft Investor.com)

However, it would be incorrect to conclude that being a component provider is necessarily better than being a system provider. The disk drive industry is renown for its cutthroat price competition. Table 3 below shows the profit performance for two representative firms in the disk drive industry. Although like Intel and Microsoft both are providers of system components, the fact that they do not own the standards led to cutthroat competition. In fact, their margins are much worse than that of Compaq, Dell, and Gateway, running as low as a quarter of the best systems manufacturer.

5 Year Averages	Seagate	Quantum
Gross Margins (%)	25.1	16.1
Pre-tax Margins (%)	3.9	2.1

Table 3 Profit Performance of Representative Firms in the Disk Drive Industry (Source: Microsoft Investor.com)

The key difference rests in Intel and Microsoft's ability to transform their components into a platform on top of which other firms built their business. Because Intel and

Microsoft sold products that were critical components of this common platform and controlled the intellectual property to prevent entry, the two firms were able to extract enormous profits. On the other hand, Seagate and Quantum's products were never transformed into a platform. Moreover, their products conformed to open standards, such as IDE or SCSI, and hence can be easily replaced.

8.3 Possible Strategy for the Provider of UBA

Based on the discussions above, one possible strategy is to follow in the steps of Intel and Microsoft and provide the UBA as an open platform to other firms in the industry with the goals of achieving rapid dominance and ownership of the standard.

8.3.1 Achieving Rapid Dominance

In order to achieve rapid dominance, it is critical to license the UBA without discrimination and for a low-to-nonexistent fee. Specifically, the provider should license the UBA's interface, or its API, by providing the binary version of the implementation freely. However, unless a non-competitive agreement has been secured, the provider should not release the source code of the implementation. This strategy enables other firms to build services that interconnect with and discourages new entrants from competing against the UBA.

Historically, this was the same tactic Microsoft undertook. For example, to compete against Netscape's Web server, Microsoft gave away its own version, the Internet Information Server (IIS), by bundling it with Windows NT. More importantly, Microsoft created a similar but separate interface for IIS that was incompatible with that of Netscape's. This had several important outcomes. First, software can be written for IIS easily taking advantage of all the services Microsoft provided. Second, Microsoft did not have to worry about additional competition against its Web server as a result of providing the API. Third, software written to support IIS must be rewritten to support Netscape, and as a result, raised the switching costs for the third party software vendors. Microsoft profited from the increase in sales for Windows NT as it was needed to use IIS.

In addition, from the consumer's perspective, the services provided by the UBA can be characterized as experiential information goods. Information goods tend to have high initial fixed costs of production and low to non-existent marginal costs of distribution. Software and content-based products, such as music CDs and home videos, are examples of information goods. Given this particular cost structure, the provider must increase the quantity demanded in order to spread the fixed cost over a large base. Since it costs virtually nothing to provide the service to the marginal consumer, the provider wants to sign up as many consumers as possible. In the long run, this allows the provider to gain a large installed-base of customers, which is a valuable asset in itself as discussed earlier, and builds momentum behind the standard.

Building momentum behind the standard is critical to the success of the UBA due to the network externalities that exist in such products. Network externality describes the concept that a product is more useful to a consumer if more people consume the same product. For example, most users want to buy the same word processor or spreadsheet package as everyone else, because this facilitates the exchange of files. In general, "the purchase price of software is minor in comparison with the costs of deployment, training, and support.... Consumers [are] much more worried about picking the winner of the [standards war] than they [are] about the price" [VS99]. By gaining build rapid momentum, the provider creates the image that the service is indispensable.

Lastly, from the perspective of other firms in the industry, control over a large installed base of users is equally valuable. Most software companies first release products for the Microsoft Windows platform before releasing a similar version for the Macintosh operating system, because Windows users outnumber Mac users by at least 10:1. By providing an open interface, the provider of the UBA guarantees access to a large number of users and will encourage other companies to support the platform. This allows the provider to build a network of offerings to attract even more consumers and corporations.

8.3.2 Profiting from the Control of the UBA Platfonn

There are many different ways the provider of the UBA can profit from its investment.

Control of a valuable distribution channel enables an innovative provider to profit from the sale of complementary products. In the retail industry, companies often use loss leaders to attract traffic and profit from cross selling. If the provider of the UBA is a financial institution, it can now pitch its own products to users as appropriate. On the other hand, if the provider is not a financial institution, then it can earn a referral fee whenever it refers customers to another financial institution.

In addition, Web portals today provide an alternative business model. Web portals typically give away their content in hopes of attracting a large number of viewers. Once they have a large audience, they can then resell portions of their Web page as advertising space. For example, Yahoo! provides free email, stock quote, and chat room services in hopes of selling advertisement space on each of those Web pages. Others Web-based search engines, such as Alta Visa, have the capability of selling the positioning of a particular Web page on the list of answers returned.

Lastly, in the more traditional credit card networks, Visa collects a percentage of each transaction. Likewise, the UBA can charge a small transaction fee for each transaction that passes through its network. For example, in Chapter 4, we discussed the possibility of transferring funds from one institution to another. Currently, the options range from writing a check and dropping it in the mail to wiring the funds electronically across the ACH network. The former is inexpensive but slow while the latter is prohibitively expensive for small amounts. A recent survey shows that banks charge approximately $30-50 for wire transfer services.

The opportunities are limited only by one's imagination.

In summary, firms must have "(1) control over an installed base of users, (2) intellectual property rights, (3) ability to innovate, (4) first-mover advantage, (5) manufacturing abilities, (6) strength in, and (7) brand name and reputation" [VS99].

CHAPTER 9: FUTURE WORK

The concepts and results that we have documented in this thesis represent initial, exploratory work in an area of on-going research. Although the prototype constructed demonstrates several interesting and powerful possibilities for the UBA, it also uncovered many areas that require additional improvements. For example, we would like to integrate the UBA with different security mechanisms in use today as well as provide users with the capability to conduct transactions. In addition, we believe the technology developed while building the UBA can be applied to other applications of information integration and would like to pursue those further.

In this chapter, we will present some suggestions for future research. In Section 9.1, we focus on some of the application level opportunities. In Section 9.2.2, we focus on the technology-specific issues.

9.1 Applications

9.1.1 Virtual Bank

To the casual visitor, Virtual Vineyard (http://www.virtualvineyard.com) looks just like any of the other wine producers one would expect to find over the Internet. In addition to allowing visitors to buy wines and specialty food from their Web site, they run a monthly wine program and make recommendations on wines according to the occasion. However, hidden behind the façade of its beautiful icons and shopping cart technology is an entirely different sort of company. Unlike its more traditional brethren, Virtual Vineyard does not grow its own grapes or own the oak barrels and machinery used to produce its own private label. Instead, it runs a completely virtual enterprise, hiring others to manufacture its products.

Virtual Vineyard's success at transparently outsourcing, or unbundling, the manufacturing of its wines from the retail sales and recommendation services it provides suggests that one might be able to repeat the same strategy for different industries. Independent money managers, accountants for small businesses, or financially savvy individuals can all create their own virtual bank by providing their clients with the best possible selection of financial products manufactured by different financial institutions.

For example, a recently developed usage scenario for the UBA demonstrates how an investor can borrow low cost capital from credit card issuing banks that offer attractive introductory rates, deposit the cash received in a high interest bearing account, and benefit from the interest rate spread. As new offers appear, the UBA can notify the manager promptly.

In addition to helping managers optimize the portfolio of investments by identifying comparable products that earns a higher return, the UBA can also provide a useful interface to hide the complex machinations from the customers. For example, the customer may simply want to know the amount of cash equivalents he/she has available and not the names of the ten different bank accounts where the money is deposited. This allows the customers to havc unencumbered access to their financial portfolio but also allows the managers to shift assets from one financial institution to another to take advantage of the latest offers.

We believe some, if not all, of these services are currently available through private banking services. However, private bankers typically limit their client base to those people whose assets exceed a million dollars. We feel it would be an interesting challenge and a useful demonstration of our technology to create a virtual bank for retail customers that duplicate some of the services available through private banking today.

9.1.2 Information Portals

The Web has been compared to many different things in the past, ranging from the information superhighway to something akin to the dial tone for delivering applications across the network. However, the truth of the matter is most people regard the Web as an enormous distributed database. The Web's success stems partly from the wide range of information it contains, ranging from the schedules of moving showings at the neighborhood theater to data on the average annual rainfall around the world. In the past couple of years, various "portal" Web sites have sprung forth to provide a central location for aggregating information, the most famous of which is the My Yahoo service. Unfortunately, such portal sites are designed for human consumption. Computer programs cannot comprehend and benefit from the collection of information.

We believe one of the important next-steps in the near future will be to provide the same type of information access to computer programs that the Web has made available to humans. We envision a world of many information servers that cater to computer programs, making information and calculations available on demand. In the same way that the Domain Name Server (DNS) provides naming services to computer programs that need to resolve an Internet domain name into a set of IP addresses, information servers will do the same for other sorts of information, such as exchange rates. With these services in place, a Web page for a hotel in Geneva will be able to easily show a Japanese visitor the prices in Japanese yen instead of simply redirecting the user to an exchange calculator Web site as they currently do.

By providing a large number of wrapped, integrated, and mediated sources to the public, we would encourage the development of applications that leverage such services and better understand people's needs and requirements.

9.2 Technologies

In this section, we describe some of the technologies that we should enhance to support the applications described above and extend the functionality of the UBA.

9.2.1 Transactional Capabilities

The current implementation of the UBA prototype only provides reporting functionality. However, as we described in Chapter 4, reporting is only half of the story. Not only does adding transactions enrich the feature set and the number of possible scenarios, transactions, including ACH wire transfers between accounts and bill payments through the credit card network (American Express, Visa, and MasterCard), represent a very profitable line of business. Most banks currently charge approximately 2-4% of each credit card bill to handle a transaction. CyberCash, a provider of an Internet gateway for handling Internet-based credit card payments, charges another 30 cents per transaction. Therefore, extending the UBA prototype to handle transactions may be of great interest to financial institutions that sponsor this research.

Extending the UBA prototype to incorporate transactions brings up many interesting issues. For example, how should one provide personal and portable storage of transactional data and how should the reconciliation of such data with bank account data be conducted? Moreover, we should also identify the current cost structure in the existing networks for handling transactions and determine if we can build a cheaper and more efficient replacement.

9.2.2 Security Mechanisms

Currently, the UBA prototype is not concerned with the details of security. As a result, it is only capable of accessing data sources that do not require user identification. In order to allow the UBA to access password protected data sources, such as credit card accounts and brokerage statements, we have duplicated the data source by caching the information locally. This allowed us to separate the task of accessing the data source into two components, namely satisfying the authentication requirements and understanding the content provided. In the near future, we would like to replace our current cached data sources with real time information.

9.2.3 Use of OFX instead of CSML

The current implementation of the prototype does not use the upcoming financial industry XML standard called OFX (Open Financial Exchange). Rather, we have decided to create our own specification for handling credit card statements. Our choice allows us to demonstrate the ideas without becoming mired in the complexity of OFX. However, as we discussed in Chapter 5, OFX has a wide ranging of sponsors, ranging from Citibank in retail banking to Charles Schwab in stock brokerage business. Although we do not expect the findings to be different, in order to ensure that the UBA prototype closely resembles an actual implementation, we may want to replace the use of CSML with OFX. Moreover, switching to an OFX-based implementation may allow us to better leverage the existing tools and benefit from the ongoing development in the industry.

9.2.4 Automatic Context Mediation Via Declarative Syntax

Our familiarity with the context of the data sources used in the examples and our knowledge of their intended application allowed us to simplify the problems of context mediation mentioned earlier. Context mediation was handled directly in the server code as the data stream from each source was parsed. However, this tight coupling of data source and application is not always feasible. Hence, our current solution is not applicable for general use.

Ideally, we would like to uncouple the context mediation of a data source from the application that uses the data. This provides us with two distinct benefits. First, in a distributed environment where the providers of information are often different from the consumers of information, the application developer will not be familiar with the domain of the information's context. Hence, uncoupling of application and data source is necessary if the architecture is expected to have a widespread usage. Second, successful uncoupling requires that we define a general framework for specifying context information This framework can be leveraged in several additional ways, including the ability for the query planner to optimize the execution of the query.

BIBLIOGRAPHY

[Bosak97] Bosak, Jon, "XML, Java, and the Future of the Web", *World Wide Web Journal,* 2-October-1997.

[BHL98] Bray, Tim, Hollander, Dave, and Andrew Layman, "Namespaces in XML", *World Wide Web Consortium Working Draft*, 27-March-1998.

[BPS98] Bray, Tim, Paoli, Jean, and C. M. Sperberg-McQueen, "Extensible Markup Language (XML) 1.0", *W3C Recommendation*, 10-February-1998.

[BS96] Bowers, Tab and Marc Singer, "Who will Capture Value in On-line Financial Services?", *The McKinsey Quarterly*, Number 2, 1996.

[Carey99] Carey, Theresa W., "Growing Pains", *Barron's*, 15-March-1999.

[DF98] Deutsch, Alin, Fernandez, et al., "XML-QL: A Query Language for XML", *Submission to the World Wide Web Consortium,* 19-August-1998.

[Gomez99] Gomez Advisor's Web Site
http://www.gomez.com

[HHH97] Hagel III, John, Hewlin, Todd, and Todd Hutchings, "Retail Banking: Caught in a Web?", *The McKinsey Quarterly*, Number 2, 1997.

[Loizos98] Loizos, Constance, "Taking on AutoDesk," *Red Herring*, November-1998.

[Madnick99] Madnick, Stuart, "Metadata Jones and the Tower of Babel: The Challenge of Large-Scale Semantic Heterogeneity", Sloan School of Management, Massachusetts Institute of Technology.

[Maier98] Maier, David, "Database Desiderata for an XML Query Language", *Submission to the World Wide Web Consortium,* 19-August-1998

[MMS98] Moulton, Allen, Madnick, Stuart and Michael Siegel, "Context Mediation on Wall Street", *COOPIS 98.*

[Rein99] Rein, Lisa, "The Quest for an XML Query Standard", *XML.COM*, 02-March-1999.

[Qu96] Qu, J. F., "Data Wrapping on the World Wide Web", *Technical Report CISL WP #96-05*, Sloan School of Management, Massachusetts Institute of Technology.

[SAB99] Sahuguet, Arnaud and Fabien Azavant, "Building light-weight wrappers for legacy Web data-sources using W4F".

[SAL99] Sahuguet, Arnaud and Fabien Azavant, "Looking at the Web through XML glasses", *Submission to CoopIS'99.*

[SAW99] Sahuguet, Arnaud and Fabien Azavant, "Web Ecology: Recycling HTML pages as XML documents using W4F", *Submission to WebDB'99.*

[Shah98] Shah, Syed Ahmed Zaman, "Design and Architecture of the Context Interchange System", *CISL WP#98-05*, Sloan School of Management, Massachusetts Institute of Technology.

[VS99] Varian, Hal and Carl Shapiro, *Information Rules: A Strategic Guide to the Network Economy,* Harvard Business School Press, 1999.

[Walsh98] Walsh, Norman, "A Guide to XML", *World Wide Web Journal*, 2-October-1997.

[W3C98] World Wide Web Consortium (W3C), "The Document Object Model", 1998.

[Zeichick99] Zeichick, Alan, "XML X-plained", *Red Herring*, February-1999.