

Chapter 4

INTERNET ARCHITECTURES FOR APPLICATION SERVICE PROVIDERS

Borko Furht, Chris Phoenix, John Yin, and Zijad Aganovic

Abstract

In recent years, business on the Internet has exponentially increased. Consequently, the deployment and management of business applications on the Internet is becoming more and more complex, which requires the development of new Internet architectures suitable to efficiently run these business applications. In this chapter we present and evaluate several computing models and related Internet architectures for application service providers. We also introduce the server-based model and the corresponding Internet architecture and two case studies, which use the proposed architecture for application deployment.

1. INTRODUCTION

1.1 COMPUTING MODELS FOR INTERNET-BASED ARCHITECTURES

The increasingly competitive, global marketplace puts pressure on companies to create and deliver their products faster, with high quality and greater performance. To get the new products and technologies to consumers is through a new industry called Application Service Providers (ASPs). Similarly to Internet Service Providers, that linked businesses and consumers up to the Internet, ASPs lease software applications to businesses and consumers via the Internet. These applications range from word processing programs to payroll management software, document management systems, and many others. The major challenge is to develop an efficient Internet-based architecture, which will efficiently provide access to these software applications over the Internet.

Application architectures have traditionally followed software development architectures. The software development architectures can be classified into:

- Traditional desktop computing model
- Client-server computing model

- Network computing model
- Server-based computing model

Traditional desktop computing model assumes that the whole application is on the client and the application is executed locally. The client must be a fat client.

Client-server computing model assumes that clients are powerful and processing is centered around local execution on clients. Computer resources were split between a server and one or several clients. This architecture allowed for larger, more scalable applications to be brought to a larger number of clients. However, the key for this architecture was to successfully partition the complexity of overall application and determining correctly which part should reside on the server and which part should run on the client. As more and more functionality migrated to the client, it became harder for applications to be maintained and updated.

Network computing model, supported by Sun, Oracle, Netscape, IBM, and Apple, assumes that software applications are dynamically downloaded from the network into the client for execution by the client. This architecture requires that the clients are fat.

Server-based computing model, supported by Citrix, assumes that business applications reside on the servers and can be accessed by users without requiring them to be downloaded to the client. The client can be either thin or fat.

In the proposed Internet-based architecture we selected server-based computing model, which is described in detail in the following section.

1.2 SERVER-BASED COMPUTING MODEL

The fundamental three elements of the server-based computing model are [1]:

- Multi-user operating system
- Efficient computing technology
- Centralized application and client management

Multi-user operating system allows multiple concurrent users to run applications in separate, protected sessions on a single server.

Efficient computing technology separates the application from its user interface, so only simple users commands, received through keystrokes, mouse clicks, and screen updates are sent via the network. As a result, application performance does not depend on network bandwidth.

Centralized application and client management allows efficient solution of application management, access, performance, and security.

Server-based computing model is very efficient for enterprise-wide application deployment, including cross-platform computing, Web computing, remote computing, thin-client device computing, and branch-office computing, as illustrated in Figure 1 [1].

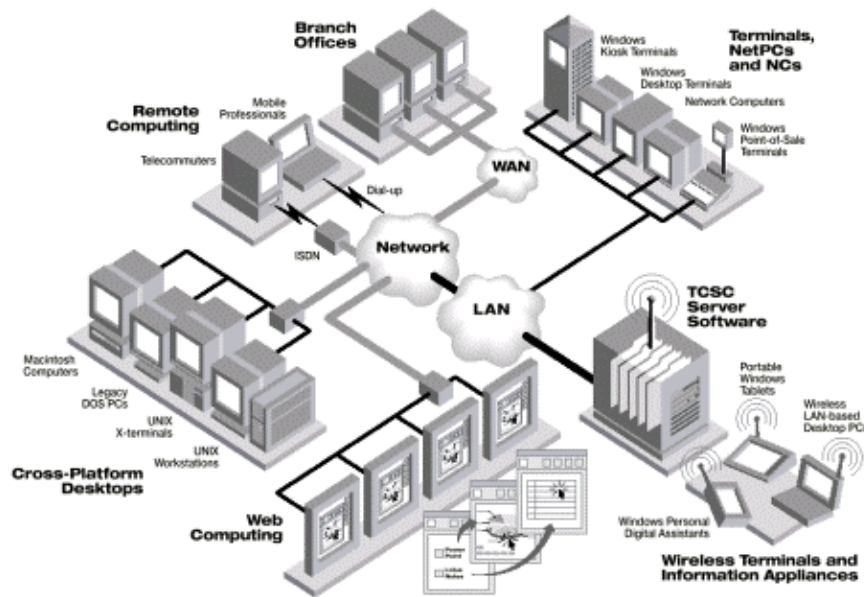


Figure 1. Server-based computing model can be used for enterprise-wide application deployment.

2. EVOLUTION OF INTERNET-BASED APPLICATION SERVICE ARCHITECTURES

Similarly to software development architectures, applications service architectures have emerged from the traditional client-server architectures to three-tier and multi-tier architectures.

The first generation of Internet-based application service architecture was based on delivery of information via public Web sites. This technology, sometimes referred to as the “first wave” Internet [2] employs the Web to present the information to the user and then allows the user to give some relevant information back. The primary focus of this architectural model is mass distribution of public information over the Internet. This architecture, which focuses on accessing information, consists of three levels (or three tiers) – presentation level, content level, and data and service level, as shown in Figure 2 [2].

At the presentation level, there is the client system, which is used to view Web page information. The client contains both presentation and application logic components. At the content level, there is a Web server that provides interactive view of information from a relational database. Finally, at the data and service level, there is a relational database system, which provides data for the Web server. This architecture is also called three-tier architecture consisting of client tier, Web server tier, and database tier.

With the advancements of the Internet, the Web, and related technologies (such as Java and HTML), as well as acceptance of standard communication protocols (such as TCP/IP and HTTP), a new architecture has emerged. In this architecture, sometimes referred to as the “second wave” Internet [2] or network-based application architecture [3], focus is on highly targeted, private distribution of software services over Intranets and Extranets. In this architecture, the Web page is not only the agent for providing information, but it also offers a

variety of application services to speed up business transactions and offer additional services. This architecture consists of n-tiers and offers maximum functionality and flexibility in a heterogeneous, Web-based environment. An example of four-tier architecture is shown in Figure 3.

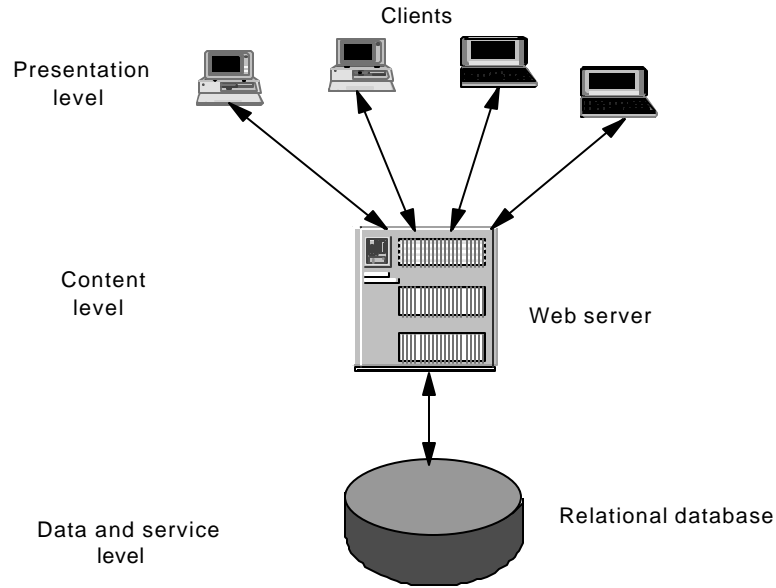


Figure 2. The three-tier architecture for application service providers is focussed on accessing information.

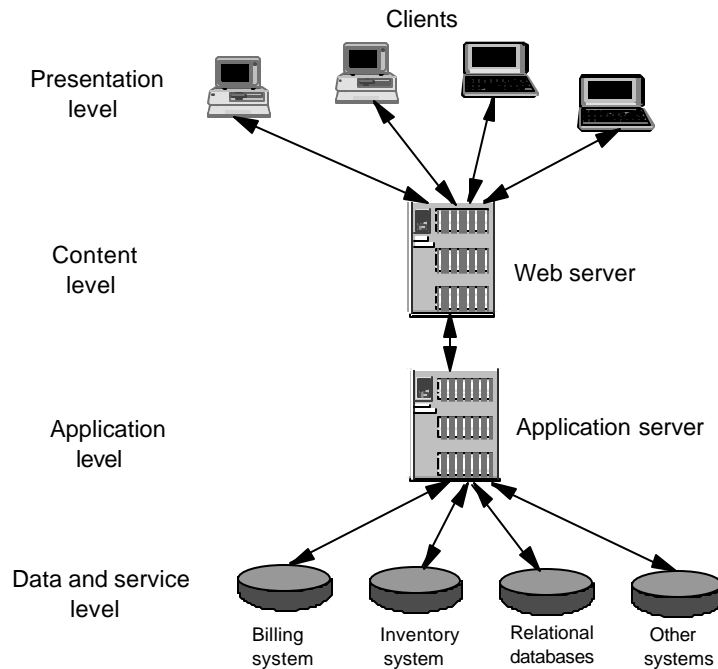


Figure 3. The multitier Internet-based architecture for application service providers is focussed on accessing application services.

At the presentation level, the client views Web pages for information as well as for a variety of application services. At the second, content level, Web server provides interactive view of information and supports client-initiated transactions. At the third, application level, there is an application server, which is used to find requested data and services, makes them available for viewing, and carries out transactions. At the fourth, data and service level, there is a variety of data and services accessible by the application server. This architecture, also called multitier architecture, consists of client tier, Web server tier, application server tier, and database tier.

Two-tier Internet architecture is typically limited for systems with a small number of users, a single database, and non-secure network environments.

3. APPLICATION SERVER

In the second generation of Internet architectures, the focus has shifted to access to business services rather than to information only. The main component of the system is an application server, which searches for services and data – this is done in the background without involving the user.

The main challenges in developing the first generation of Internet architectures and application services were related to user interfaces and cross-platform interoperability. In developing the second generation of Internet architectures, the main challenge for service developers is to deliver their services seamlessly via the Internet, which in turn requires innovations in many areas. The following challenges need to be addressed in developing the second generation of Internet architectures:

Standards. There are many standards for developing Web pages, which causes difficulties for developers.

Increased programming complexity. The implementation of business services on the Internet is very complex programming task.

Network performance. Business applications over Intranets and Extranets require very reliable and high performance networks.

Security. Business applications on the Internet require a very high level of security.

Web access to legacy applications. As mentioned earlier, the new Internet architectures is focused on accessing various business applications rather than just information.

Database connection support across Web-based requestors. Users should be able to access a variety of databases connected to the application server.

The majority of these functions, sometimes called *middleware*, are implemented in application servers that provide support for developing and deploying business applications located on the server or partitioned across client and server.

Application server offers support for developing and deploying business logic that may be located on the server or, more often, partitioned across client and server. Running business applications on the server provides many benefits [4].

3.1 KEY TECHNOLOGIES FOR APPLICATION SERVERS

Key technologies for developing contemporary application servers include:

- Java programming language and environment,
- JavaBeans – the Java based component technology, which allows the development of new applications more rapidly and economically,
- ActiveX – the competing technology to JavaBeans, which is Windows platform-dependent and language-independent,
- Java Database Connectivity (JDBC) – the Java SQL that provides cross-platform database access for Java programs,
- Java servlets – small Java routines that service HTTP requests and dynamically generate HTML,
- CORBA – provides a standard architecture for distributed computing and interoperability on the Internet

The key technologies are described in details in Chapter 10.

Java application servers have recently emerged as an efficient solution for the application server tier. A Java application server provides the following features:

- Make it easy to develop and deploy distributed Java applications.
- Provide scalability, so hundreds to thousands of cooperative servers can be accessed from ten of thousands clients. Therefore, Java must be fully multithreaded and have no architectural bottlenecks that prevent scaling.
- Provide an integrated management environment for comprehensive view of application resources (for example, Java Beans, objects, events, etc.), network resources (databases), system resources (ACLs, threads, sockets, etc.), and diagnostic information.
- Provide transaction semantics to protect the integrity of corporate data even as it is accessed by distributed business components.
- Provide secure communications.

CORBA (Common Object Request Broker Architecture) and JavaBeans are open standards for component software development and deployment that allow writing small code objects that can be reused in multiple applications and updated quickly. They also allow developers to expose legacy system data and functionality as services available over the Web, and therefore most application servers are based on these technologies.

For example, the CORBA architecture makes it possible to find and use services over the Internet. Similarly, Enterprise JavaBeans is a standard sever component model for Java application servers that provides services to network-enable applications, so that they may be easily deployed on Intranets, Extranets, and the Internet [5].

CORBA provides universal connectivity in broadly distributed environments as well as cross-platform interoperation of both infrastructures and applications. The object Web model based on CORBA and other standards is shown in Figure 4 [7].

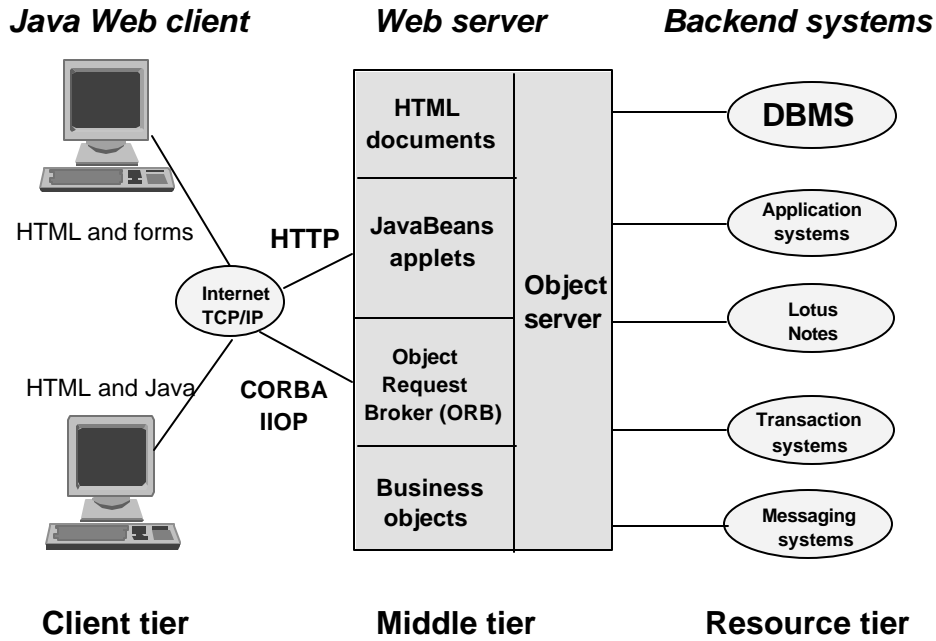


Figure 4. The object Web-model based on CORBA and other standards provides universal connectivity in distributed environments.

CORBA currently provides many services including naming, security, transactions, and persistence, as illustrated in Figure 5 [7].

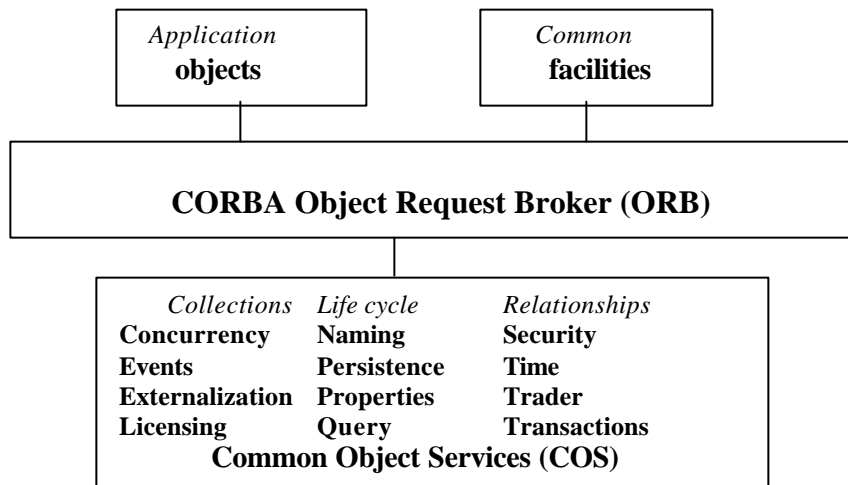


Figure 5. CORBA provides a standard for interoperability that includes many services required by object applications.

4. IMPLEMENTATIONS OF INTERNET ARCHITECTURES FOR APPLICATION SERVICE PROVIDERS

In this section we present four popular architectures for application service providers (ASP) developed by Sun, Netscape, IBM, and Microsoft.

4.1 SUN'S ARCHITECTURE

Initially, Sun Microsystems defined in Fall 1996 Java-based application development architecture, which consisted of three tiers: the client tier that provided user interface, the middle tier for business logic and database access, and the database tier, as illustrated in Figure 6 [8].

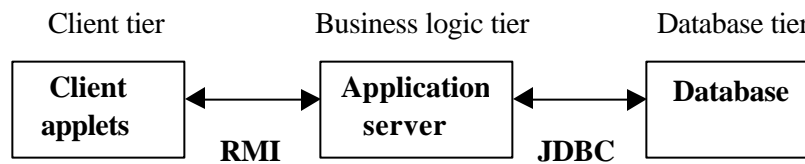


Figure 6. The Sun's Java-based three-tier architecture for ASP.

Sun selected Java language for the client tier, which provided more sophisticated GUI capabilities than HTML implementation. Client applets did not perform significant business logic functions in order to keep clients as thin as possible. Java technology was also used for the middle tier and the middle-tier servers were implemented as standalone Java applications.

Because both client and middle tiers are implemented using Java, client – middle tier communication was performed using Remote Method Invocation (RMI), where the middle-tier servers created the necessary RMI objects and made them available to clients via the RMI object registry. The middle tier communicated with the database via the JDBC API. This architecture is based on client/server computing model, in which client resides on the user's desktop, and the middle and database tiers reside on one or more of five data centers around the company.

Recently Sun has developed an enhanced multitier architecture, which included an additional tier – the WebTop server tier, as shown in Figure 7 [8].

In the three-tier architecture (Figure 6), applets were dynamically downloaded at runtime to the users' locations from an application server. For remote locations and modem connections with constrained bandwidth, applet download time was a few minutes, which was unacceptable.

Another issue related to three-tier architecture was the access to network resources such as files and printers. Java prohibits applets from accessing any local or network resources. In addition, Java does not allow communications with any machine other than the one from which the applet was downloaded. As a result of these limitations, file access occurred at the middle tier. This meant that information might be sent from the client to the middle tier and then back to a file server near the client.

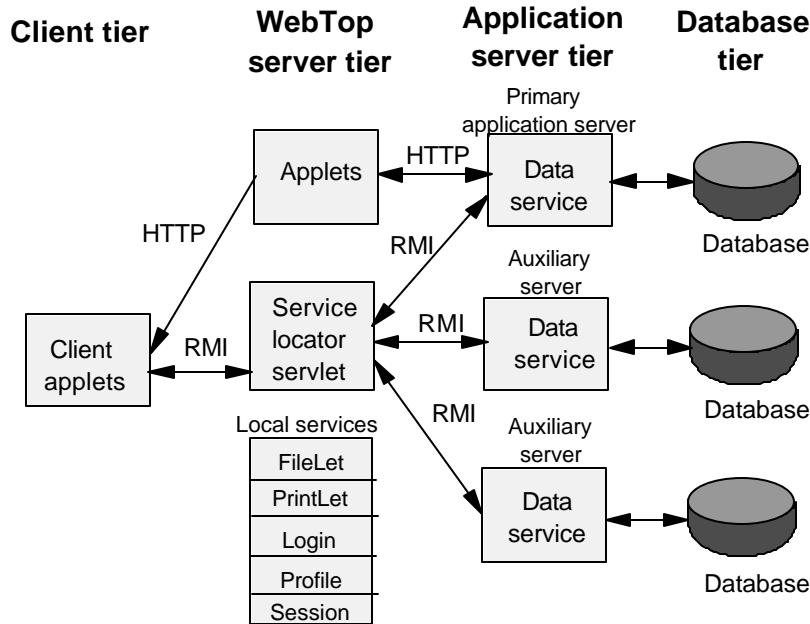


Figure 7. The Sun's Java-based multitier architecture for ASP.

Introducing a new tier, WebTop server tier, has resolved the issues related to the three-tier architecture. The WebTop server runs the Java Web server and is located near the users it serves. This server is used as a cache for applets and static application data, so the first problem was resolved. The server also supports services that access network resources such as user files and printers, which are typically located near the users. Finally, the WebTop server is used to find the services that users need.

In the architecture in Figure 7 the client is thin and typically includes a graphical user interface written as an applet that runs from a Web browser. The application server tier provides access to data and implements business logic and data validation. The application server is responsible for all database transaction handling.

The communication between the client and WebTop server tier and between the WebTop server and the application server tier, HTTP and RMI are used. Communications between application servers and databases is performed via JDBC.

One of the main benefits of the multitier architecture is that it increases application scalability and performance by enabling clients to be connected concurrently. In a client-server model clients are directly connected to databases, while in a multitier architecture only application servers connect directly to databases. In this way, the application server can multiple requests from many clients through a pool of preallocated database connections, thus reducing the database server load. Load on the application server tier can be balanced by using multiple application servers.

Another benefit of the multitier architecture is that it supports thinner clients, because most of the logic runs in the application server and database tiers. Thus broad range of client platforms can run the applications.

4.2 NETSCAPE'S ARCHITECTURE

Similarly to Sun's architecture, Netscape recently developed multitier architecture for application service providers, which is based on the separation of presentation logic from application logic, as illustrated in Figure 8 [2].

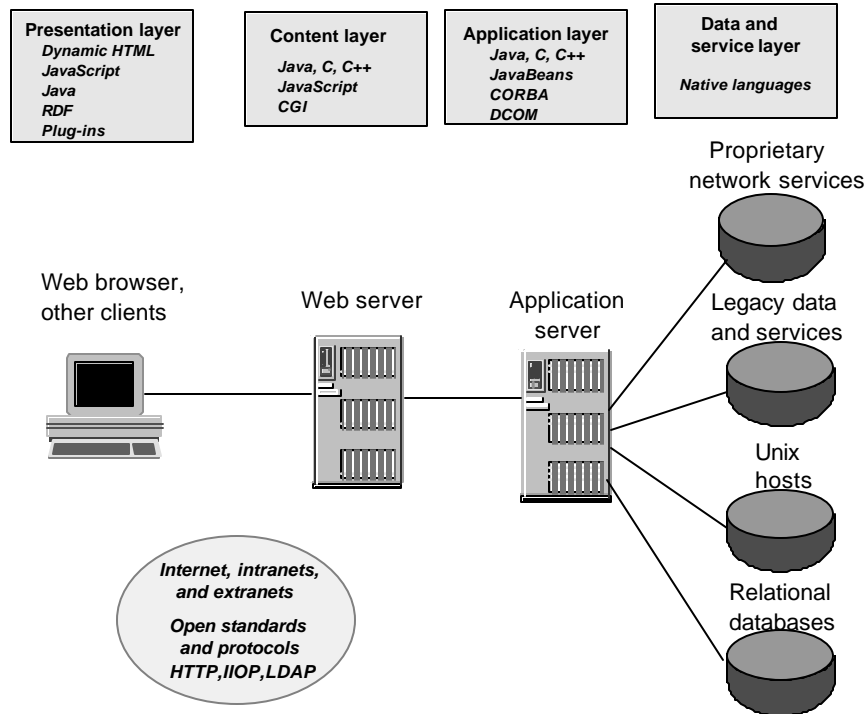


Figure 8. Netscape's multitier architecture for ASP.

In the Netscape's multitier architecture, the client tier is typically based on an open-standard browser such as Netscape Navigator. The presentation logic and GUI is built using HTML pages that include Java applets. At the content level, a Web server primarily uses HTTP. It provides base-level information and content services as well as simple database information access via Java, JavaScript, and other high-level CGI scripting languages like Perl.

Application server uses CORBA and JavaBeans components or objects. Transaction services enable access to relational databases and other legacy systems.

First three levels in the multitier architecture in Figure 8 are provided by Netscape technologies and products, while the last level - back-end services and other legacy systems are accessed through standard Internet interfaces.

4.3 IBM'S ARCHITECTURE

IBM has developed the Component Broker, which is Internet middleware for distributed objects [7]. Component Broker is a software system that allows developers to build, run, and manage Web-enabled business objects, components, and applications. Component Broker consists of:

- Tools for building distributed and business objects, and applications,
- A runtime that provides a distributed-object infrastructure on the middle tier, and
- A system management functions for the distributed object runtime and its resources.

Component Broker architecture, shown in Figure 9, accepts inputs from any clients (Java or C++) transported via Internet InterORB Protocol, and ActiveX transported via a bridge. The object server consists of components that provide control, services, context, and connection resources.

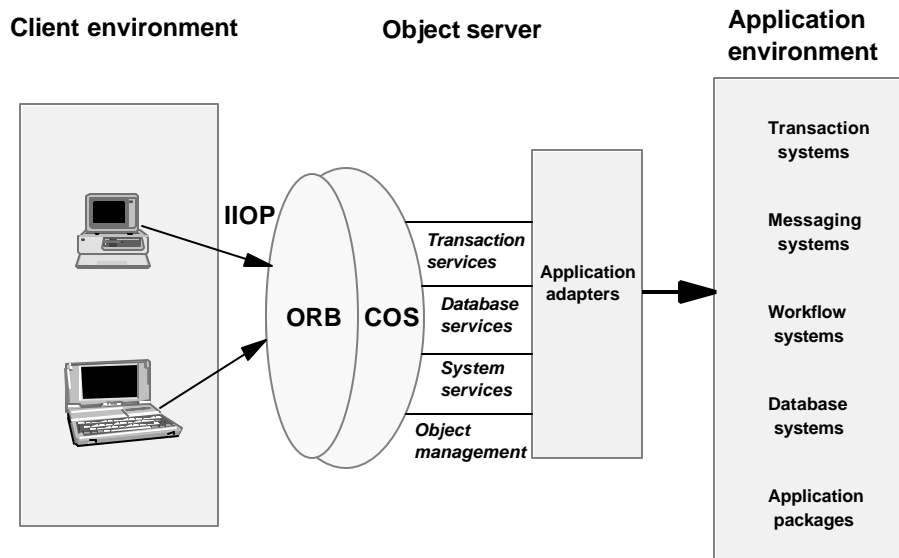


Figure 9. Architecture of the IBM's Component Broker at the middleware tier.

The Component Broker receives client requests through the CORBA-compliant Object Request Broker (ORB). Object services are supplied through the CORBA Common Object Services (COS). These services provide object transaction services, database services, system services, and object management functions, as illustrated in Figure 9.

Application adapters connect Component Broker object applications with existing software systems and applications.

4.4 MICROSOFT'S ARCHITECTURE

Microsoft Internet architecture is a component-based architecture based on Windows DNA [14]. The heart of Windows DNA is the Component Object Model (COM) that allows developers to build applications from binary software components at any tier of the application architecture. These components provide support for packaging, partitioning, and distributing application functionality [14]. Distributed COM (DCOM) enables communications between COM components that reside on different machines. DCOM is a competing model for distributed object computing to CORBA, described in Section 3.

5. A CONTEMPORARY ARCHITECTURE FOR ASP

In this section, we propose ASP computing architecture using server-based computing model and ASP application architecture.

5.1 ASP COMPUTING ARCHITECTURE

Our computing architecture for application service providers is based on the server-based computing model, described in Section 1.1. As we indicated earlier, in server-based computing all applications and data are managed, supported, and executed on the server. This architecture provides the following benefits:

- Single-point management
- Predictable ownership costs
- High reliability
- Bandwidth-independent performance
- Universal application access
- Use of thousands of off-the-shelf applications
- Low-cost and fast application development
- Use of open standards
- Graphical and rich user interface
- Wide choice of client devices

The proposed server-based architecture uses two technologies developed by Citrix:

- Independent Computing Architecture (ICA), and
- Windows-based terminal (WBT)

Independent Computing Architecture is a Windows presentation services protocol that turns any client device (thin or fat) into the thin client. The ICA consists of three components: server software, client software, and network protocol.

On the server, ICA separates applications from the user interface, while on the client users see and work with applications' interface. The application logic executes on the server. The ICA protocol transports keystrokes, mouse clicks, and screen updates over standard protocols requiring less than 20 Kbps of network bandwidth.

A *Windows-based terminal* is a thin-client hardware device that connects to Citrix server-based system software. The WBT does not require downloading of the operating system or applications and there is no local processing of applications at the client, as in the case of other thin clients such as network computers or NetPCs. A WBT has the following features:

- An embedded operating system such as DOS, Windows CE, or any real-time OS
- ICA protocol to transport keystrokes, mouse clicks, and screen updates between the client and the server
- Absolute (100%) execution of application logic on the server
- No local execution of application at the client device

The proposed architecture also allows consumers and business to access software applications from their Internet browsers. This is provided using Citrix's software *Charlotte*. In addition, software component *Vertigo* allows more interactive applications on the Web. This software allows customized Web pages such as electronic trading accounts to be updated automatically without hitting the refresh button on the computer.

The proposed architecture for ASP using server-based model and Citrix technologies is shown in Figure 10.

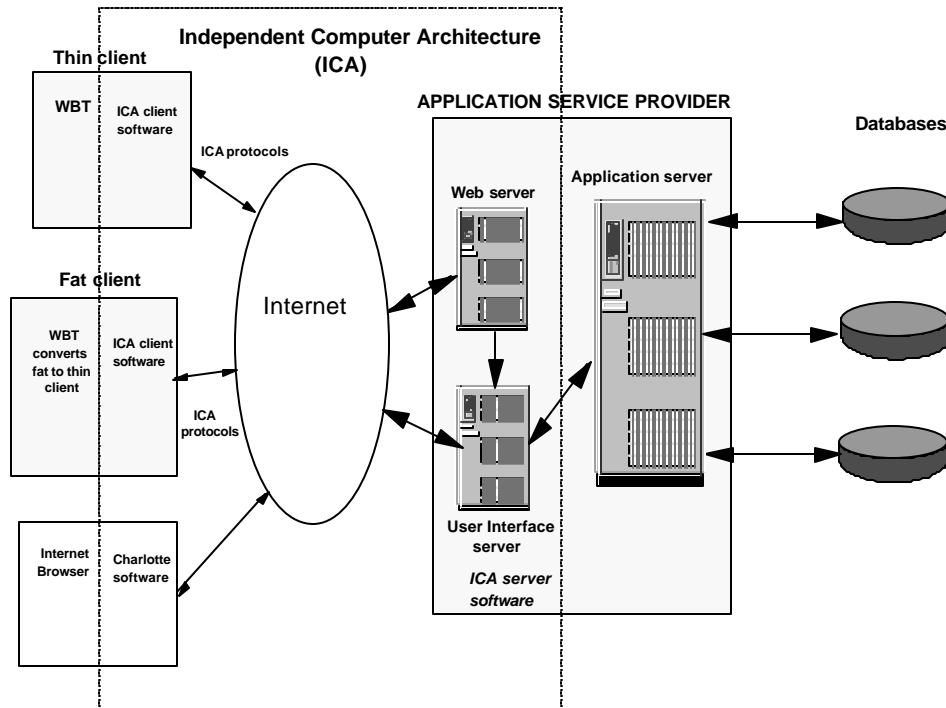


Figure 10. The proposed architecture for ASP uses server-based model. All applications are executed at the server or cluster of servers.

The proposed architecture is platform independent and allows non-Windows and specialized ICA devices to run Windows applications residing and executing on application server farm. Application server farm is a group of application servers that are linked together as a single system to provide centralized administration and scalability.

The architecture in Figure 10 allows application service providers to rapidly develop and deploy applications across complex computing environments. It also provides application access to all users, regardless of their location, type of client device, or form of network connectivity. The architecture can be applied to any type of client hardware, and therefore requires no change in client hardware. The system significantly reduces requirements for network bandwidth compared to other architectures. Finally, the proposed architecture reduces the total cost of application, as analyzed in Section 6.

5.2 ASP APPLICATION ARCHITECTURE

To take maximum advantage of ASP computing architecture, a new breed of applications needs to be developed. The key drivers of new distributed application architecture is a need for wide spectrum of thin clients, bandwidth usage optimization, application multi identity shared back end computing, reliable data flow management, security, legacy application integration, and long list of service operation requirements. The diagram shown in Figure 11 can depict a desired architecture of an ASP application.

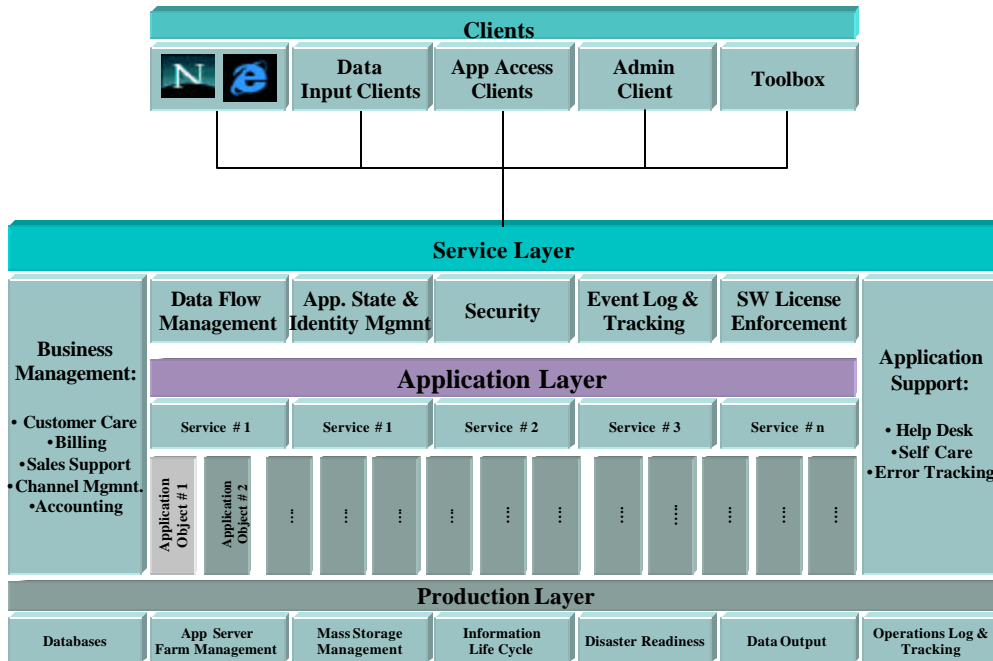


Figure 11. Architecture of an ASP application.

5.2.1 Client Software

ASP application client software is in general very different from types of client software provided as part of traditional client server applications available on the market today. To support ASP business model, client software must be “thin” requiring minimum computing power, installation and support effort, minimum communication bandwidth, and minimum version upgrade. Highly distributed nature of ASP service requires from client software ability to support versatile data inputs, highest level of user’s security, and ability to support multiple communication protocols.

Data Input

ASP service architecture is in essence remote computing architecture, which requires capabilities to generate and import application data into the remote application. Data can be generate as part of specialized batch program or as by-product of third party software. Data input clients can stand alone or integrated within other clients or legacy applications. Multi step data flow requires advanced information security, tracking, reporting, and above all ability to restore data in case that any stage system failure results in data loss. Data input clients may or may not be “thin.” Footprint of these clients is primarily defined by local functionality necessary to create the data at optimum cost.

Application Access

Application access clients are characterized by limited local computation capability and remote command capability to the server side application concentrated at service back end. These clients are the ones that should be as generic and as “thin” as possible. Smaller and simpler the client lesser the operational cost at the front end. The ideal application access client is plain Web browser. However, browser access is limited to very low level of functionality provided by HTML protocol. Function rich application computing requires specialized client software or plug-ins providing access to remote application at the back end.

Toolbox

To bridge the existing legacy applications with ASP service an ASP application software requires comprehensive set of APIs or application enabling tools providing the system integration capabilities and customizations.

Administration

This client should provide end user with ability to completely control its own application. Desired functions are: adding new users, setting up security profiles, managing application specific variables, usage tracking and reporting, and billing presentments and reporting.

Security

Client software security capability must include ability to authenticate users on the front end and to create virtual private channel of communication with the service back end.

5.2.2 Service Layer

Server side application is characterized by concentration of all computing and data intensive processes at back end, application multi-identity, sophisticated data flow management, and by its ability to integrate with business management, application support, and service production components. The ultimate goal of such application engineering is to create fastest computing environment, economy of scale through all customers sharing of common computing and data management infrastructure, and maximum operational readiness.

Application Layer

At the core of service layer is the application layer of software providing actual computing application packaged as specific service, for example: Service #1. This service application can be either stand alone application or user interface into integrated solution based on several other independent third party applications.

Data Flow Management

Data generated through data input clients is managed by data flow management software. One can consider this software component as data switch capable of accepting data input, decompressing and decoding data, identifying the owner of data and target data base, importing data in target data base, caching and mirroring data at each stage for disaster readiness reasons, and creating logs for data input tracking and reporting.

Application State and Identity Management

ASP provider will have many different applications for many different customers simultaneously. Also, each individual application will have many different users requiring different application set-up and profile. Application state and identity management software acts as application switch identifying individual users and applications and then assigning appropriate user's profile. Therefore, ASP application must support multiple identity capability. Ability to share the same computing and data management resources between many different users and applications is essential for reliable service delivery and economy of scale.

Business Management

ASP application should also integrate into business management software enabling automatic account creation, and usage data feed into billing solution.

Application Support

ASP application should also integrate with application support solution that consists from customer self support site.

6. EVALUATION OF VARIOUS ASP ARCHITECTURES

Analysts and IT professionals have developed numerous models for estimating the total cost of IT services, sometimes called “total cost of ownership” (TCO). In the past, these models had the hardware-centric view because they analyzed the costs of owning and maintaining desktop computer hardware. In the age of the Internet, Web-based computing, and E-commerce, applications must be accessible across a wide variety of connectivity options, from low-speed dial-up connections to wireless, WAN- and Internet connections. A contemporary cost analysis should consider the total cost of application ownership (TCA), rather than the total cost associated with specific computing devices. The Tolly Group has developed a model for comparing the TCA of different computing models, discussed earlier [9]. We present and discuss their results in this section.

In order to determine the cost of application deployment, four computing models introduced in Section 1 can be analyzed from the following points of views:

- Physical location of the application
- Execution location of the application
- Physical location of data, and
- Location of the user and means of connectivity

The cost of complexity of deploying and managing an application strongly depends on physical location of the application. The cost of application distribution, installation, and managing of updates must be considered.

The choice of where an application is executed determines the hardware, network, and connectivity costs. An application can run on the server, on the client, or in a distributed server/client environment. In some cases, the application must be downloaded from a server to a client, which has an impact on performance and productivity.

The location of stored data determines the speed at which information is available. It also has an impact on the cost related to protecting and backing up critical corporate data.

Location of the user and the means of connectivity also have an impact on the cost and complexity of deploying an application.

Table 1 summarizes the application deployment characteristics for four computing models introduced in Section 1 [9].

Table 1. Computing Models and Application Deployment Characteristics

	APPLICATION LOCATION	APPLICATION EXECUTION	DATA LOCATION	USER ACCESS	NETWORK REQUIREMENTS
Traditional desktop	Client	Client	Client	Local	None
Client-server	Client and server	Client and server	Client and server	Lan, WAN, Internet	High bandwidth
Network-based	Server	Client and server	Server or client	LAN, WAN, Internet	High bandwidth
Server-based	Server	Server	Server	LAN, WAN, Internet	Low bandwidth

Tolly Group has analyzed and calculated the total cost of application ownership for a medium-size enterprise of 2,500 users, with 175 mobile users working on the road. The calculated costs were divided into (a) Initial (first-year) cost (which includes hardware, software, network infrastructure, and user training) and (b) annual recurring costs (which includes technical support and application maintenance). The results of analysis are presented in Figure 12.

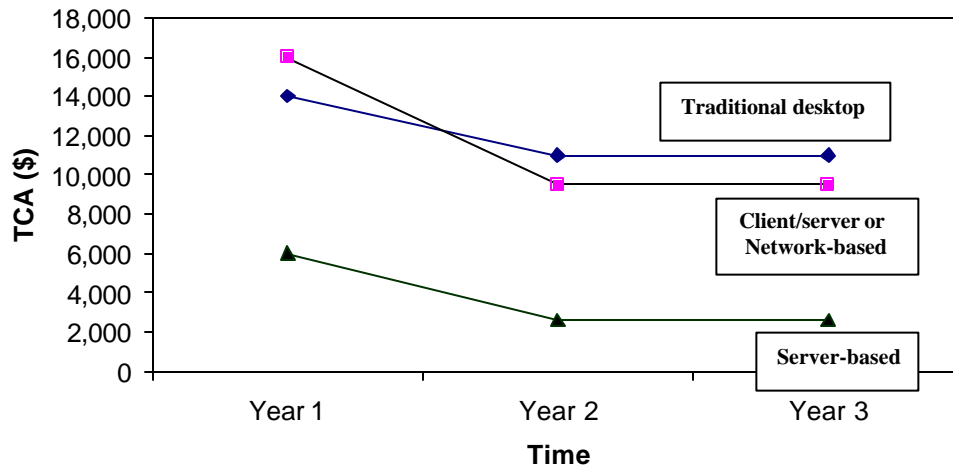


Figure 12. Analysis of total cost of application (TCA) for various computing approaches [9].

Traditional desktop computing approach requires relatively high initial cost for hardware, software, network infrastructure, and training (\$14,000) as well as very high annual recurring costs for technical support and application maintenance (\$11,000 annually).

Client/server and network computing approaches require slightly higher initial investment (\$16,000) in order to replace existing client hardware, however annual recurring costs are reduced (\$9,500). This model becomes less expensive than the traditional desktop model from the third year forward.

Server-based approach gives the best TCA both in terms of initial costs and annual recurring costs (\$6,000 and \$2,600, respectively). The reason for it is that this model allows any type of client to access any application across any type of connection. This model also provides single point for the deployment and management of applications.

In summary, the server-based model, which was applied in our architecture, is the most efficient and cost-effective solution to application deployment and management.

7. OVERVIEW OF INTERNET BUSINESSES AND ASPs

In this section we present the Internet business model, adopted from [10], and present two case studies of application service providers based on the proposed server-based ASP architecture.

7.1 INTERNET BUSINESS MODEL

In the Internet business model proposed in [10], Internet businesses are categorized into (a) horizontal businesses (or industries) and (b) vertical businesses (or industries), as illustrated in Figure 13.

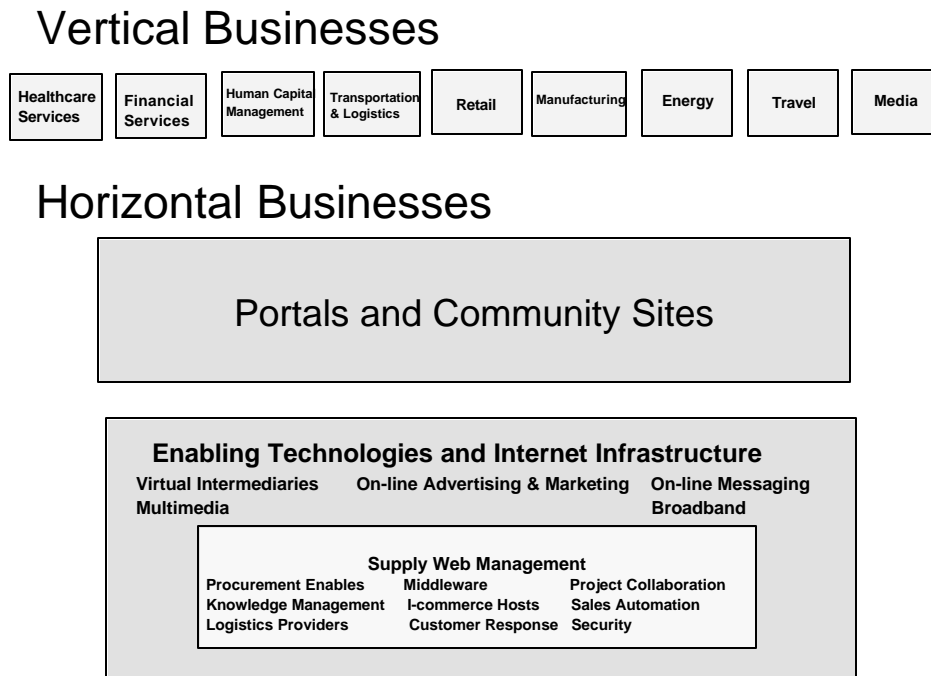


Figure 13. Internet business model consists of horizontal and vertical businesses (Source: Stephens Inc. [10]).

Horizontal industries are further divided into: (i) portal and community sites, and (ii) enabling technologies and Internet infrastructure. Some leading companies from both categories, vertical and horizontal, are presented in Tables 2 and 3 [10].

In 1998 there were approximately 28 million households in U.S. that used the Internet. According to Forester Research, this number will double in year 2003 and reach about 53 million households. Percentage of total households using E-commerce in 1998 was 29% (or 8.7 million households), and the projected growth is that this percentage would reach 35% (or 40 million households) in year 2003. Business-to-business Internet commerce in 1998 was \$43 billions, and projected market for 2002 is \$1,330 billions.

Table 2. A Sample of Vertical Internet Business Leaders [10]

VERTICAL INDUSTRY	BUSINESS LEADERS
Healthcare services	WebMD, Healtheon, Prescription.com, Soma.com
Financial services	E*Trade, Ameritrade, Knight/Trimark, NetBank, Telebank, InsureMarket, Quotesmith
Human Capital Management	Monster.com, Taps.com, NetStaff, Jones Intl
Transportation and Logistics	National Transportation Exchange, Cargo Finder, Eurotrans
Retail	Amazon.com, Buy.com, Onsale.com, CDNow, Reel.com
Manufacturing	VerticalNet, Chemdex, ProcureNet, PartNet
Energy	Altra Energy, Silicon Energy, Southern California Gas
Travel	Amadeus, Galileo, Sabre, Preview Travel, Travelocity
Media	CMP Media, CNet, EarthWeb, Thestreet.com, Sportsline

Table 3. A Sample of Horizontal Internet Business Leaders [10]

HORIZONTAL INDUSTRY	BUSINESS LEADERS
<i>Portals & Community Sites</i>	AOL, Yahoo!, Excite, Netcenter, Koz, Lycos
<i>Enabling Technologies and Internet infrastructure</i>	
Virtual Intermediaries	EBay, uBid, Island
On-line Advertising and Marketing	DoubleClick, Flycast, NetGravity
On-line Messaging	Message Media, USA.net, Critical Path, Bigfoot
Multimedia	Real Networks
<i>Supply Web Management</i>	
Procurement Enablers	Ariba, Commerce One, Dynamic Web, Supply Works
Middleware	Iona, Bea Systems, Inprise
Knowledge Management	BroadVision, OpenText, Momentum, CyLex
Project Collaboration	Microsoft, IBM
i-commerce Hosts	Globix, Exodus, Psinet
Sales Automation	Open Market, Digital River, BroadVision
Customer Response	Edify, Acuity, Aptex, netDialog
Security	Check Point, Entrust, Network Associates
Payment	Checkfree, Cybercash, Wave Systems
Logistics Solution Providers	I2 Technologies, Manugistics, Logility
Broadband	Aware, AtHome

7.2 CASE STUDIES

We describe two case studies of using the proposed architecture for ASPs:

- (a) Online document management system created by CyLex Systems and Pride Enterprises and applied for collection and management of motor vehicle applications in State of Florida, and
- (b) Billing and customer care system developed by Daleen Technologies.

7.2.1 Online Document Management System – CyLex Systems

The Department of Highway Safety and Motor Vehicles (DHSMV) serves as the central point of collection and management for all motor vehicle title applications generated throughout the state. DHSMV needed to replace its existing title microfilming operation. The current system was costly, slow, and did not provide the level of service expected by the state's citizens. Simple requests for title applications could take up to three weeks.

CyLex Systems, Inc. has teamed with its business partner, Pride Enterprises, to propose a totally outsourced solution. Pride Enterprises performs all document conversion and indexing, while CyLex Systems, Inc. provides a complete on-line document information management system. This solution reduces title retrieval time to 15 seconds once the documents have been processed.

The CyLex Systems, Inc. outsourcing solution provides the following benefits:

- Reduces the cost of document management
- Reduces document retrieval time from days and weeks to minutes or seconds
- Requires no up-front investment in technology or hardware
- Requires no on-going cost of maintaining and updating technology
- Is quickly implemented
- Is complementary to the existing computing infrastructure

Overview

Located in Tallahassee, Florida, the State of Florida Department of Highway Safety and Motor Vehicles is one of the four largest motor vehicle agencies in the country. DHSMV receives approximately 18,000 title applications per day consisting of 4.5 pages each for a total of 81,000 documents. This represents a yearly total of approximately 20 Million documents. In such a vast, diverse and demanding environment, the ability to provide consistently high quality administrative services while containing costs presents an ongoing challenge.

The Micrographics Section of the Division of Administrative Services of DHSMV was performing these tasks with a staff of 48 full-time employees. Staying up to date with such a consistently high volume was technically impossible; backlogs often approximated 17 days. For example, document retrieval was stretched beyond an acceptable time frame, taking anywhere from days to weeks to retrieve a title.

The CyLex Express® Solution

DHSMV issued a Request for Proposal (RFP) for document image management services to replace the department's existing microfilming process. DHSMV challenged contractors to

provide a turnkey solution that would effectively manage the millions of documents generated every year. Contractors were also evaluated based on the quality of their technical proposals, implementation and administrative costs. To meet the needs of DHSMV's RFP, CyLex Systems, Inc. and Pride Enterprises proposed a document image management system with four functional pieces consisting of document preparation, image capture, image administration and image review and retrieval, as shown in Figure 14.

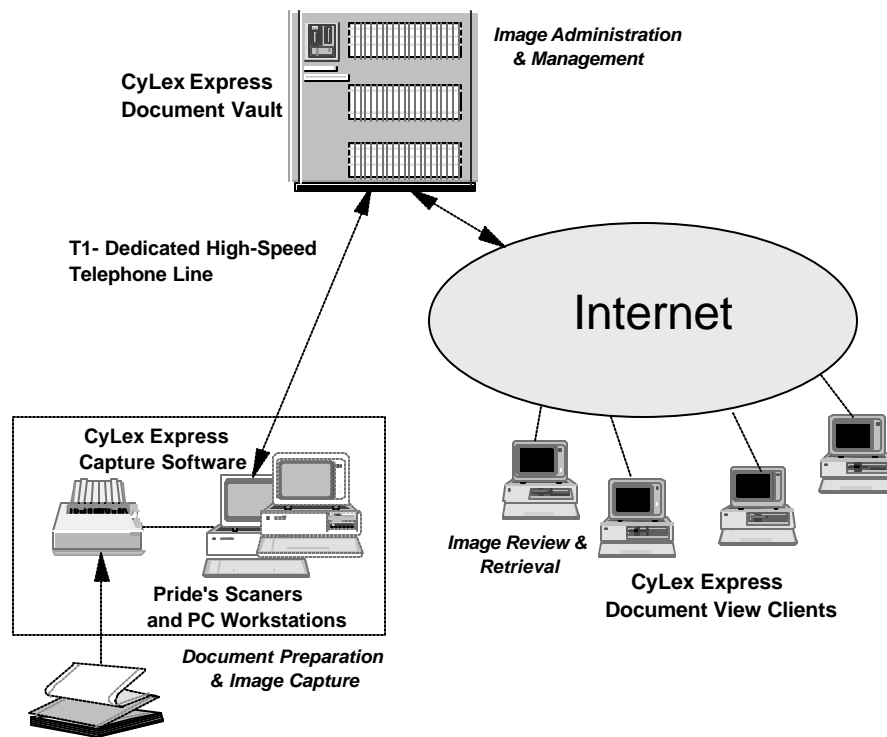


Figure 14. Document on-demand image management system applied at DHSMV.

The CyLex Express Service provides a centralized and secure document vault for storing and managing electronic images of paper files. The *CyLex Express Capture Software* provides all the functions needed to scan, index, and upload the documents. *CyLex Express View Software* allows remote viewing of documents as well as printing.

The DHSMV project consists of high volume document scanning and indexing. Pride Enterprises performs daily document pick-ups from DHSMV headquarters in Tallahassee, Florida, and transports them to one of Pride's two secure document conversion facilities to prepare for scanning and indexing. To ensure the highest level of quality control, all indexes are double-keyed and images are examined for quality control.

Pride uses a network of high-speed scanners and document-imaging workstations to provide image capture of both fronts and backs of documents. Pride Enterprises' equipment includes: two Kodak 9500D machines, capable of scanning 180 pages per minute; one Kodak 3500 capable of scanning 175 pages per minute, and 12 indexing quality control PC workstations. In addition, Pride Enterprises maintains file servers and communication gear with a T1 dedicated high-speed telephone line for the fastest possible data transmission to the *CyLex*

Express Document Vault. The T1 line is capable of handling 1.54 megabits of data per second.

All document content is managed by the CyLex Document Vault, a state-of-the-art server facility, maintained by EMC Corporation, the world's leading supplier of intelligent enterprise storage and retrieval technology, located in Hopkinton, Mass. On-line documents are managed on RAID media and near-line in tape libraries. Document access is enabled only to authorized DHSMV employees.

DHSMV uses the CyLex Express Software for on-line electronic retrieval of documents. Scanned images are stored and managed by the CyLex Express Document Vault. DHSMV accesses the CyLex Express Document Vault through direct dial up or an Internet connection.

An important feature to DHSMV is the built-in layers of security that CyLex Express Software integrates into the system to protect documents and to ensure that only users with the proper authorization privileges have access to the CyLex Express Document Vault. Security features include a firewall preventing illegal entry; a multi-tier log-in ID/password procedure that is designed to ensure that each user can access only documents they are authorized to use; and positive client authentication. Optionally, CyLex Express allows all scanned documents to be encrypted at the client site and then stored.

Currently, DHSMV maintains 25 computer workstations that are authorized for users of CyLex Express with direct access to the Document Vault. The vault also provides real-time document retrieval of on-line stored images within the 15-second-per-image retrieval time required, anytime, 7 days a week.

Phase II of the DHSMV project will provide direct on-line access to the vault from more than 200 local tax collector offices statewide. On-line access will enable them to view 4.5 million title applications filed each year in Florida, as well as 18 million additional images of documents. This will translate into a reduction of administrative time from present 2-3 weeks using a microfilm system to 15 seconds with CyLex Express.

7.2.2 Billing and Customer Care (Daleen Technologies)

The growth of E-commerce, emergence of ASP, and ever increasing competition among the Internet-based service providers significantly complicated the Internet business models and raised the end users' expectation. Internet business models widely used include:

- Free
- Advertisement based
- Transaction based
- Utility based (usage + recurring fee).

At the same time, end users no longer want to be treated as mass. Instead, they want to be treated as individuals and communities [17]. These new requirements and challenges helped to push billing and customer care (BACC) to the top of prioritization list of any serious Internet-based service provider such as an ASP and to propel BACC vendors such as Daleen Technologies to stardom. H. Adams put it very well: "Without Billing, It's Just a Hobby." [18].

The BACC functions required by a typical ASP include:

Billing

- Pricing
- Service bundling and discounting
- Usage collection
- Usage rating
- Taxing (future)
- Payment processing
- Treatment and collection
- Settlement and commission processing
- Service Level Agreement (SLA) penalty processing

Customer Care

- Customer management
- Help desk
- Service ordering
- Electronic bill presentment and payment
- Online self-care

As we can see, the BACC components hold critical information (customer data, service data, and usage data) for a service provider. Using such data, a service provider can determine its most valuable customers from the rest, offer superior personalized customer service/support, tailor its services to an individual or a community, and conduct revenue planning and assurance.

On the other hand, BACC is only one of many business support systems (BSS) and operation support systems (OSS) that an ASP may need to have. Other systems include network management, security, accounting, asset management, and decision support systems. Therefore, a BACC must have necessary interfaces (such as APIs) to support interoperability with other BSS/OSS systems so that they can access BACC data and functions. Figure 15. shows how Daleen Technologies' BillPlex™ uses an N-tier architecture to satisfy such requirements.

In addition to such an advanced architecture, the next generation ASP BACC solution must include the following capabilities:

1. Ability to collect (possibly through a usage mediation product) and rate usage data related to applications and services offered by the ASP.
2. Ability to “personalize” billing for an individual user of a community of users so that the pricing and billing are value- rather than usage oriented.

Several companies have shipped products that help to collect usage data for ASPs. XACCT, Narus, and Softblox have products that can collect IP and application (such as Microsoft Word) usage data. An industry workgroup (<http://www.ipdr.org>) is also working on a standard for IPDR (IP Detail Record Format). One of the objectives for this standard is to uniformly represent usage data for both IP and ASP services.

In a customer-centric environment, which is widely believed to happen, pricing and billing need to be “personalized” for customer based on the value provided to the customer and NOT on the resource used. Because value is perceived differently by each customer, value-based pricing/billing requires thorough understanding of the customer. Such understanding not only helps pricing and billing for existing services but also enables a service provider to define new tailored services for the customer. This positive feedback cycle will continue to push

customer satisfaction and service provider revenue higher. Only a right BACC system can make this happen.

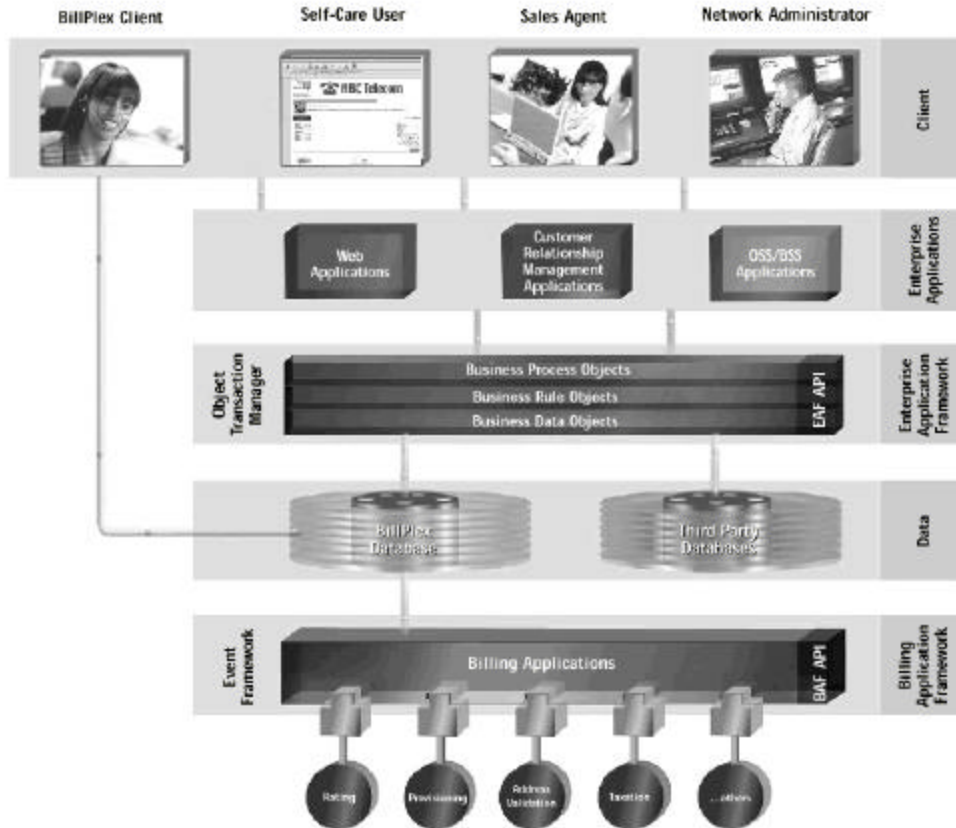


Figure 15. The N-tier architecture used by BillPlex.

Interestingly, not only ASPs need BACC functions, BACC functions can also be provided through an ASP (i.e. *BACC ASP*). In fact, billing service bureaus have long been in existence. Some of these billing service bureaus used mainframe and remote terminals to provide billing services to telephone and utility companies. However, a new generation of BACC ASPs will emerge to offer far superior BACC services through the ASP model. Using Internet and leading BACC products such as BillPlex, these BACC ASPs will be able to bring the best BACC solution to their customers at the least cost.

8. CONCLUSIONS

In this chapter we presented and evaluated contemporary multi-tier Internet architectures, which are well suited for application service providers. We evaluated several computing models for Internet-based architectures and proposed a server-based computing model, which has a number of advantages over the other models. We also presented two case studies of Internet applications that use the proposed ASP architecture.

References

1. "Server-Based Computing," Citrix Systems, white paper, www.citrix.com, 1999.
2. P. Dreyfus, "The Second Wave: Netscape on Usability in the Services-Based Internet," *IEEE Internet Computing*, Vol. 2, No. 2, March/April 1998, pp. 36-40.
3. "Software Development for the Web-Enabled Enterprise," Sun Microsystems, white paper, 1999.
4. "What is a Java Application Server," BEA WebLogic, weblogic.beasys.com, 1999.
5. A. Thomas, "Selecting Enterprise JavaBeans Technology," WebLogic, Inc., Boston, MA, July 1998.
6. R. Orfali, D. Harkey, and J. Edwards, "Instant CORBA," John Wiley & Sons, 1997.
7. C. McFall, "An Object Infrastructure for Internet Middleware: IBM on Component Broker," *IEEE Internet Computing*, Vol. 2, No. 2, March/April 1998, pp. 46-51.
8. Gupta, C. Ferris, Y. Wilson, and K. Venkatassubramanian, "Implementing Java Computing: Sun on Architecture and Application Development," *IEEE Internet Computing*, Vol. 2, No. 2, March/April 1998, pp. 60-64.
9. "Total Cost of Application Ownership," The Tolly Group, Manasquan, NJ, White paper No. 199503, June 1999.
10. J.B. Eichler, R.Y. Roberts, K.W. Evans, and A.L. Carter, "The Internet: Redefining Traditional Business and Giving Rise to New Ones," Report, Stephens Inc., Little Rock, AR, May 1999.
11. D. Rosenberg, "Bringing Java to the Enterprise: Oracle on Its Java Server Strategy," *IEEE Internet Computing*, Vol. 2, No. 2, March/April 1998, pp. 52-59.
12. M. Benda, "Internet Architecture: Its Evolution from an Industry Perspective," *IEEE Internet Computing*, Vol. 2, No. 2, March/April 1998, pp. 32-35.
13. "Enterprise JavaBeans Technology: Server Component Model for the Java Platform," Sun Microsystems, white paper, java.sun.com, 1999.
14. G.R. Voth, C. Kindel, and J. Fujioka, "Distributed Application Development for Three-Tier Architectures: Microsoft on Windows DNA," *IEEE Internet Computing*, Vol. 2, No. 2, March/April 1998, pp. 41-45.
15. C.J. Woodard and S. Dietzen, "Beyond the Distributed Object Decision: Using Components and Java Application Servers as a Platform for Enterprise Information Systems," *Distributed Computing*, 1998.
16. G. Pour and J. Xu, "Developing 3-Tier Web-Based Enterprise Applications: Integrating CORBA with JavaBeans and Java Servlets," *Proceedings of the 3rd International Conference on Internet and Multimedia Systems and Applications*, Nassau, Bahamas, October 1999.
17. L. Downes and Chunka Mui, "Unleashing the Killer App," Harvard Business School Press, 1998.
18. H. Adams, "Communications Billing & Customer Care: Time to Think Outside the BOCs," *Impact!*, August 1999.