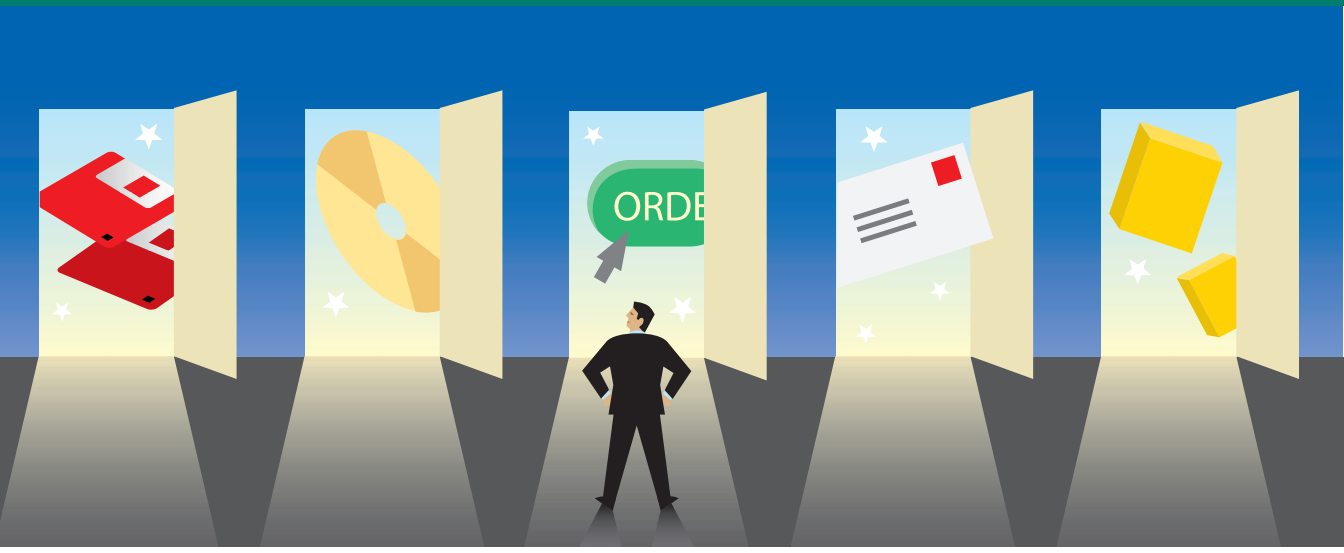
Exploring IBM e-Business Software



Become an Instant Insider on IBM's Internet Business Tools

"The most complete and insightful coverage of [IBM's software] portfolio I've ever read...

— Bruce R. Williams, Briefing Manager, SVL Executive Briefing Center, IBM

Casey Young

Table of Contents

Introduction xvi
What This Book Is xvi
What This Book is Not xviii
How to Use This Book xviii
Your "Members Only" Web Site xix

Part One: The Framework Around Your e-Business

Chapter 1: IBM and e-Business 3

What is e-Business? 4
Why is e-Business Important? 6
What is e-Commerce? 7
Business-to-Business (B2B) 8
IBM Framework for e-Business 9
Framework Programming Model *13*
Clients *15*
Web Application Server 17
External Services *20*
Infrastructure *20*
WebSphere Software Platform *22*
Concepts to Make an e-Business Work 24
Tag Languages *26*
Java *29*
Summary 31
Case Studies 32
Pristine Appliance Company *32*
Pegasus Airline *32*
Definitions 33

Chapter 2:
Architectures that Pre-Date e-Business **38**

Batch 39
VSAM 41
OLTP 43
Databases 45
Distributed Architectures 49
Object-Oriented Databases 52
Hybrid Database Management Systems 54
Component Broker Architecture (COBRA) 56
Business Intelligence 58
Applications 63
Summary 63

Chapter 3:
Hardware Architectures and Operating Systems **65**

Hardware 66
The New Series 71
Hardware Architectures 72
Operating Systems 73
Mainframe 73
Non-Mainframe 76
The World is Blending 78
Operating Systems Summary 80
Mainframe Centric 80
Non Mainframe Centric 80
Network 81
Summary 84
Definitions 84

Part Two: IBM Framework for e-Business: Build, Run, Manage

Chapter 4:
Building Your e-Business Application **89**

Presentation Tools 92

WebSphere Portal Server *93*
WebSphere Personalization *96*
WebSphere Transcoding Publisher *98*
WebSphere Voice Server *101*
Websphere EveryPlace Suite *103*
Development Tools 107
VisualAge for Java *108*
VisualAge Generator *113*
WebSphere Homepage Builder *115*
WebSphere Studio *117*
WebSphere Business Components *120*
IBM SanFrancisco *124*
WebSphere Edge Server *128*
Websphere Site Analyzer *129*
WebSphere Host Integration *130*
Summary 132
Tech Stuff 133
Case Studies 135
Omni Bank *135*
Timeless Skiing *136*
Definitions 137

Chapter 5: Software Products for Running Your e-Business 141

Storing Data 143
IMS (Information Management System) *143*
DB2 Universal Database (DB2 UDB) *150*
DB2 UDB for OS/390 ..... *154*
DB2 UDB for Windows, UNIX and OS/2 (DB2 UDB UNO) *159*
Content Manager *163*
Information Movement 166
MQSeries Messaging *167*
MQSeries Everyplace *171*
MQSeries Integrator *171*
MQSeries Workflow *172*
DB2 Connect *173*
Data Replication Solution *173*
DB2 DataJoiner *174*
DB2 DataPropagator *174*
IMS DataPropagator *175*

IBM DataRefresher *175*
Web Application Servers 176
Lotus Domino *177*
WebSphere Application Server *180*
Enterprise Information Portal 184
Summary 185
Tech Stuff 186
IMS *186*
DB2 Family of Products *186*
Content Manager *187*
MQSeries *187*
Data Replication Solutions *187*
Lotus Domino *188*
WebSphere Application Server *188*
Enterprise Information Portal (EIP) *189*
Case Studies 189
Saturn Drug Store *189*
ABC Insurance Company *190*
Definitions 190

Chapter 6: Managing Your e-Business Application 193

Background 196
The Problem 197
The Privacy Concern 198
Application Management Specification 200
Tivoli Module Builder Suite 203
Tivoli Module Builder *203*
Tivoli Module Designer *204*
Tivoli Enterprise 206
Tivoli Management Framework Architecture *206*
Tivoli Management Framework Physical Structure *209*
Toolkit *210*
Availability Management 211
Tivoli Service Desk *211*
Tivoli Decision Support *212*
Monitor Enterprise 212
Business Systems Management *213*
Tivoli Enterprise Console (TEC) *213*
Prevent Unavailability 214
Tivoli Distributed Monitoring *214*

Storage Area Network (SAN) Solution 215
Tivoli Storage Network Manager *215*
Tivoli Storage Manager *216*
Tivoli SANergy *217*
Tivoli Services *217*
OS/390 Resource Measurement Facility (RMF) *218*
System Automation for OS/390 ..... *219*
Tivoli Application Performance *219*
Tivoli NetView for OS/390 ..... *220*
Tivoli NetView Performance Monitor *221*
Tivoli Decision Support for OS/390 ..... *221*
Tivoli Web Management 221
Security 222
Security Concepts 224
Tivoli SecureWay Policy Director *225*
Tivoli SecureWay Privacy Manager *228*
Tivoli SecureWay User Administration *228*
Tivoli SecureWay Risk Manager *228*
Tivoli SecureWay Security Manager *231*
Tivoli SecureWay Global Sign-On *232*
Tivoli SecureWay Public Key Infrastructure *233*
Tivoli SecureWay FirstSecure *235*
SecureWay Boundary Server (SBS) *236*
SecureWay Firewall *237*
SecureWay Toolbox *238*
Summary 239
Tech Stuff 239
Case Studies 241
Terry's Trucking *241*
Xena Markets *242*
Descriptions 242

Part Three: Planning for e-Business

Chapter 7: Application Accelerators 247

WebSphere Commerce Suite 248
WebSphere Commerce Suite Start Edition *249*

IBM WebSphere Payment Manager *250*
WebSphere Commerce Suite Pro Edition *254*
WebSphere Commerce Suite Marketplace Edition *256*
WebSphere Commerce Suite Service Provider Edition *256*
WebSphere Catalog Manager 256
WebSphere BtoB Integrator 258
WebSphere/Domino Integration 259
Patterns for e-Business 260
e-Business Partner Solutions 264
Enterprise Resource Planning and Customer Relationship Management 267
Summary 268
Tech Stuff 268
Case Studies 269
High-Flying Auto Parts *269*
Challenger Technology Services *270*
Definitions 271

Chapter 8: An Approach to e-Business 273

Where Do You Want to Go? 275
Where You Are 278
IBM Sales Team 281
IBM Web Sites 281
e-Business Resource Site *282*
alphaWorks *284*
Putting the Pieces Together 285

Part One

The Framework Around Your e-Business

The first three chapters of this book are not about IBM's e-business software per se, but rather provide a structure for understanding IBM's e-business software and its positioning. Chapter 1 focuses on the IBM e-business strategy, viewed from several angles. There is the corporate strategy, as expressed by Lou Gerstner, Chairman and CEO of IBM. The implementation of that strategy in the IBM Framework for e-business is discussed next. This Framework is important because it provides a blueprint of how IBM focuses its hardware, software, and service offerings to help its customers implement e-business. The chapter also discusses an implementation of the IBM Framework for e-business: WebSphere Software Platform. This platform views e-business from a WebSphere brand perspective—the brand that provides many of the components for the run-time environment and connectors that allow e-businesses to exist. The final section of the chapter covers a few e-business concepts that are important to understanding the rest of the book.

Chapter 2 looks at the architectural developments that have led to e-business from batch processing through the object-oriented paradigm and the development of massive applications such as Siebel. These architectures are important to the development of an e-busi-

ness system because most companies will not want to rip out every process they already have in order to build an e-business system—nor can they afford to! IBM is providing ways to incorporate these existing systems into e-business systems as needed. This chapter also answers the perennial question, "Why was it we did it this way?"

One of the largest problems that companies face today with the development of a cohesive e-business strategy is the proliferation of hardware and software—some of which is compatible, some of which is not. It is one of the reasons that IBM stresses industry standards. Chapter 3 looks at one portion of that problem—hardware architectures and operating systems. This includes the physical boxes for storage, the operating systems that run them, and network concepts. While these items are covered more extensively in other books of the series, it is important to briefly review them here as a basis for e-business. With the exception of Linux, only IBM hardware and software platforms are covered.

These chapters are the foundation chapters of the book. Other chapters in the book reference concepts and strategies covered in these chapters. In essence, they are the framework for the book.

1

IBM and e-Business

This is a book about software—specifically IBM's e-business software. Software is what makes the computing world work. Hardware—the physical box that sits on your desk or in a computer room—makes nothing more than a great doorstop if the software is not there. It is software that makes it work and better software makes it work better.

If all this sounds simple, look at the Internet. Originally developed by the U.S. Department of Defense in 1969 to promote networking research, the ARPANET, as it was called then, was little more than a collection of computers linked together by wires. There was software, but it was crude and rudimentary and you really had to want to use it to invest the time it took to learn it. The first major change to the Internet was the development of a standard communications protocol (a standard software coding language) to handle communication between different components (computers and wires) on the Internet. This coding language was called TCP/IP (Transmission Control Protocol/Internet Protocol). The second major change, the change that really made the Internet a common appliance in every home was the development of the World Wide Web and its common language—HyperText Markup Language (HTML). Both of these are software developments that will be discussed throughout this book.

And what is the world's largest software company? Well, if you stood IBM's software division on its own, it would be the world's largest software company. IBM has millions of lines of software code that makes the world run. Used to thinking of IBM as a hardware company and Microsoft as the world's largest software company, many people are surprised by this fact. Not only is IBM the world's largest software company, but it is also a software company committed to e-business. They have already invested $1 billion building the e-business category within the company.

This chapter focuses on three areas about e-business: a working definition of e-business and some of its sub-components, a discussion of IBM's strategy for e-business, including the IBM Framework for e-business, and a discussion of the more common elements, such as XML, that are components of most e-business solutions.

What is e-Business?

If you ask three people what e-business is, you will get three different answers. In fact, if you ask the same person the question at three different times over the last decade, you would also get three different answers.

To many people the faces of e-business are the Web page, and the media hype of fortunes made and lost by the "dot-com" businesses. The ability to buy and sell things on the Internet was fascinating. Surely people would rush to use this new technology to do their daily shopping. The holiday season would become the poster child of the New Economy as millions clicked their way through the Web maze, ordering this and that from here and there. And for a while it seemed like that was true. Until the immutable business laws of supply and demand and return on investment reared their collective heads and caused the tumble of many "dot-coms" in 2000.

But this shakeup is only part of the picture. According to a speech given by Lou Gerstner, chairman and CEO of IBM, at the e-Business Conference Expo on December 12, 2000, this is what happens with transformational technologies. First there is wild enthusiasm, then a period of disillusionment, and, finally, reality. And what is this reality? The Internet is a tool—no more, no less. And if we treat e-busi-

ness as a business and learn how to leverage the tool, we can do very well indeed.

According to Gerstner, there are two main things that will be important for the emergence of an e-business founded on fundamentals: integration and infrastructure.

Integration is about the entire span of what happens with a business transaction, from the moment of the first click on the Web page to the final reckoning with the government tax bureau. These different business systems need to be able to interact with each other, securely passing information, massaging the information as needed and getting the information where it needs to go when it needs to get there.

And that's the easy part.

Companies are traditionally built along business lines. Applications were built within these business pipelines. The order entry system was built separately from the supply chain system. Customer accounts were separate from sales opportunity tracking. And the business leaders of these units were separate from each other and frequently ran their line of business as if it were a separate company.

For e-business to work, companies that have invested in their pipeline organizational model are going to have to make significant change to become an integrated e-business. Handshakes between one pipeline and another are going to have to be fast, secure, and reliable. For companies that have the desire to become leaders in the e-business world, IBM has the software to make that integration happen.

Infrastructure is the second important aspect of e-business. Gerstner pointed out three items that were important for e-business infrastructure. The first is that e-business infrastructure is "end-to-end." This doesn't mean that clients connect with servers, but that front-end applications access applications within the firewall that access applications that are outside the firewall that return information back into the organization. It also means that all the new devices that are becoming front-end applications need to be considered. This goes beyond receiving stock quotes on your pager. It moves to the heart of "pervasive e-business," filled with devices that talk to each other to insure that your pacemaker keeps running, your car can send a positional distress signal if you're in trouble and your refrigerator can remind you to pick up milk on the way home.

The second important aspect of infrastructure is standards. IBM is committed to the idea of open standards because in order to make

integration and end-to-end infrastructure work, components of e-business need to be able to communicate with each other in a standard manner, without a lot of adjustment. eXtended Markup Language (XML) is the software standard that is being used to communicate the meaning of all the information we pass around the globe. It is important to IBM that the XML standard remain an open standard and not become a proprietary portion of someone's software code. The Linux operating system is also important to IBM. They plan to invest nearly $1 billion in Linux in 2001. They have 1500 IBM developers dedicated to Linux-enabling their products and services.

The third important aspect of infrastructure is that it must be "self-healing." If there is a problem, be it hardware, software, or plain old human error, the software needs to be able to adapt to the problem and keep on going. If there is a spike in brokerage traffic on the day after an important economic announcement, the software needs to be able to balance that load and keep on going. If a hacker tries to break in and does some damage, the software needs to detect the break-in, heal the damage and keep on going. According to Gerstner, we are not quite there yet, but within a few years this type of "self-healing" will be commonplace in all kinds of mainstream commercial applications.

Why is e-Business Important?

In spite of the difficulties, e-business continues to grow. Why? There are two main reasons companies are turning to e-business—the number of new people on the Internet and the efficiency model of doing business on the Internet. Today, there are, by some estimates, over 330 million people in the world attached to the Internet. This group is almost evenly split between English-speaking (172.3 million) and non-English-speaking (163.7 million) users. These users jointly control about 63 percent of the world's economy. By 2003 the number of users is expected to grow to 230 million English-speaking users and 696 million non-English-speaking users, with the biggest growth in users that speak Asiatic languages (63.1 million to 250 million users). The Internet will be the most significant way to reach people who have spending power and probably political power as well.

The other significant advantage is the cost of a transaction over the Internet versus one that requires human interaction, even over the telephone. Banks determined the cost of an ATM transaction to be significantly less (about ten cents) than one involving a teller (about $1.25). This was so important to their bottom line that customers that only use ATMs for their transactions can get better rates than those whose transactions require a teller. Banking is finally moving online in a serious manner as people become more and more accustomed to doing things over the Internet—particularly their profitable customers.

These items apply to wired e-business connections, but the Internet evolution continues to wireless devices. By 2004, IDC predicts that 1.4 billion people will access Web content from wireless devices. IDC estimates that 32.5 percent of the US population use some form of wireless devices, with 52 percent of Europeans, and 60 percent of Japanese doing so. Most significant to the business planner, however, is that by 2005, it is estimated that three-quarters of all Web transactions, totaling $200 billion, will take place over the air.

What is e-Commerce?

Essentially the front-end of e-business, e-commerce is about buying and selling goods, services, and information to individuals electronically. L.L. Bean and Amazon.com are two well-known places to purchase goods. Travel agencies such as American Express and auctioneers such as e-Bay expand the concepts of e-commerce beyond the selling of goods. Charles Schwab is a leading discount brokerage company that was one of the earliest brokers to offer the ability to access accounts and place trades from anywhere, 24 hours a day, over the Internet. At the end of the last decade, almost anything imaginable (as well as some unimaginable items!) was available for purchase over the Web.

Then there are things on the Web that are less easily identified as commerce. The government puts massive amounts of information on the Web. It is probably cheaper than printing the little pamphlets that were available for order, but is it e-commerce? It is a stretch, but probably. You are just buying your information in a more circuitous

manner—through your tax dollar. And what about those banner ads on the Web? Commerce? Definitely. Especially if you click through and buy something.

Many early innovators without good solid business plans have failed as the millennium faded. Many more were plagued with problems brought on by their success—significant crashes of Web site supporting servers during peaks of high volumes, and growing customer dissatisfaction with customer service. Early reports on the 2000 holiday shopping season indicate that customer use of Web buying was up, more servers stayed up than the previous year, but that e-commerce Web sites hadn't done a good job determining how to handle returns effectively.

The future of e-commerce is good if a few fundamental practices are resolved. The first is that basic business processes must be adapted to the Web. As easy as it is to click and buy, it must be that easy to click and return, ask a question or find what you are looking for. The second large e-commerce issue is privacy.

Ironically, as people have gotten more comfortable entering their credit card information over the Web, they've become less comfortable entering their personal information. Customers will put false information in the forms that are nicely presented on the Web. (How many customers named 'Mickey Mouse' do you have in your database?) They do not trust that the company will respect their privacy and not abuse or sell their personal data. In order to combat this, companies are going to have to define clear, specific privacy policies and adhere to them.

Business-to-Business (B2B)

B2B is more about the back office applications that go beyond the company. It includes things like Supply Chain Management (SCM), Customer Relationship Management (CRM), and Enterprise Resource Planning (ERP). In addition, it provides the ability to interact with business partners and other outside agencies such as Dunn and Bradstreet and the government. They include e-marketplaces where a company can get the supplies they need "just in time" and at the best price. It can be as simple as a supplier of electrical parts selling them to a government contractor over a Web site on the Internet. It can be

more complex, with transfer of data processing files from one company to another in order to provide information about services. For example, a package shipping company could send a file of information to a company indicating the whereabouts of all the packages they had picked up for delivery that day.

IBM is working with partners to produce solutions for e-business as well as providing products for those companies that wish to build their own. These partner offerings are solutions made up of multiple components that provide ready-to-run support for applications from 20,000 business partners, including nearly 9,000 independent software vendors. In addition, for those who wish to build their own, IBM provides application accelerators such as WebSphere B2B Integrator to provide starting blocks with the tools needed to build these applications, yet based on open XML technology to allow companies to keep their options open for emerging technologies.

IBM Framework for e-Business

When IBM first began to look at e-business in the early part of the 1990s, they realized that the basics of architectural development would apply to Web development as much as it did to any other application development. To that end, they developed the Framework for e-business to serve as the core of their e-business software strategy. This Framework helps their customers build, run and manage successful e-business applications.

Figure 1.1 shows the three components of an e-business strategy resting on a foundation of operating systems. IBM software runs on all of these operating systems. This demonstrates IBM's commitment to open software, since not all of these operating systems were developed by IBM.

The software products that make up the Build component of the e-business strategy are those required to create an e-business system. These products include VisualAge for Java, Lotus Domino Designer, IBM WebSphere Studio and WebSphere Business Components. These software groups and their related products are discussed in Chapter Four.

The software products that make up the Run component of the e-business strategy are those required to create an e-business sys-

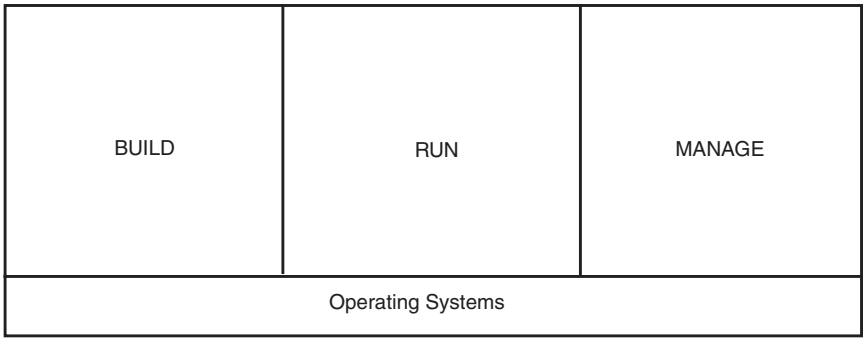


Figure 1.1. IBM Framework for e-business.

tem. These products include MQSeries, Lotus Domino, WebSphere and DB2 Universal Database. These software groups and their related products are discussed in Chapter 5.

The software products that make up the Manage component of the e-business strategy are those required to create an e-business system. These products include the Tivoli products SecureWay, Storage Management, Pervasive Management, Service Provider, and e-Marketplace Manager. These software groups and their related products are discussed in Chapter 6.

Specifically, the Framework for e-business consists of

- Principles: Standards based, open e-business foundation
- Practices: Proven development and deployment practices
- Products: Comprehensive portfolio of leadership products
- Partners: Business partner expertise and solutions

This Framework is based on multi-vendor standards like Linux, CORBA, Java, and XML. It includes the client, application server, data and infrastructure standards that make it possible for a client to access data and services anywhere in the network. This model simplifies application development and deployment and makes the most of

developer resources. Support of standards positions the Framework as an open industry supported alternative to proprietary, vendor-controlled models.

To those unfamiliar with the terms, Linux is an open-standards operating system. An operating system provides an interface between an application and the hardware that it uses. Operating systems are discussed in Chapter Three. CORBA (Common Object Request Broker Architecture) is an architecture developed by the Object Management Group (OMG) that is based on object-oriented theories. It allows pieces of programs, called objects, to communicate with each other regardless of their programming language or the operating system on which they are running. CORBA is further discussed in Chapter Two. Java is an object-oriented programming language that is portable to any operating system. Extended Markup Language (XML) is a standardized way to define the meaning of a piece of content. It is discussed later in this chapter.

In continuing the standards discussion, it is important to understand that growth of the e-business economy is dependent on the adoption of shared standards that openly define the interactions and interdependencies of the Internet. These shared, open standards underlay the initial emergence of the Internet, and will play a key role in its expansion. Shared standards enable e-businesses to optimize their business models, expand market opportunities, retain business flexibility, and improve costs. Fundamentally, however, shared standards provide businesses the ability to choose from the best information technology infrastructures, no matter what company owns them.

This only makes sense. One of the problems of being a world traveler is the necessity of carrying adapters for electronic and telephonic hook-ups for different countries or continents. But this is a minor problem compared to the chaos that would occur if every city in the world had a different adapter. This analogy applies to all of information technology development, but most especially to the Internet. If we are to get to the world that Gerstner describes, open standards must be developed and agreed to. That may take some time, but ultimately, it must prevail or we will limit our own ability. As this is being written, a significant example of this issue is being discussed in the marketplace—the ability of "instant chat" rooms to communicate with each other. Will the AOL standard, used by an estimated 80 percent of the Internet chat customers, prevail? Or will the collective standard of the other vendors prove victorious? Only time will tell.

IBM implements proven development and deployment practices by a set of set of proven, reusable architectures that guide the implementation and extension of e-business applications. These practices are referred to as IBM Patterns for e-business. They reduce a company's risk in developing new applications by matching business challenges with domain patterns, using proven application and runtime topologies, populating the topologies with pre-tested pattern mappings, and establishing the best-practice guidelines for application development.

While the Internet is a new technology, the basic business challenges remain similar. The object is to earn more than producing the good or service costs to bring to market. Within that objective there are several different methodologies that can be used. With its experience in information technology, IBM is in a unique position to understand business challenges and how to meet them from existing patterns, or domains, of the business world.

The same applies to developing application architectures. IBM is uniquely able to build from legacy systems to the new Internet structure because of its long-term engagement in the industry. Their focus was, and remains, on the inner workings of application architecture. They provide the software and processes to get the information from where it is to where it needs to be, run an application in an optimal manner and provide security and management for the application runtime environment.

One of the difficulties with testing a new application is the development of product mappings, particularly in a world where everything seems to somehow relate to everything else. IBM, through its Patterns for e-business, can give companies a starting point for this development. Discussion of these product mappings is beyond the scope of this book, but can be acquired through IBM's "red books" built specifically for each of these patterns. IBM Redbooks are a series of books (originally red!) that cover a specific aspect of product implementation. In addition to a Redbook on the Patterns for e-business, there have been Redbooks on the implementation of Siebel on DB2 UDB for OS/390, IBM's Replication Solution, implementing backup and recovery, etc. The Redbooks are available from the IBM Web site.

Consultants make a good living by understanding and applying best practices no matter what company they are in. Through experience, they have discovered what works and what does not work. This experience can be codified into a set of best practices for appli-

cation development. Best practices are part of IBM's Framework for e-business.

These Patterns for e-business are organized by the following application functionality:

- User-to-Business: self-service applications where users are interacting with enterprise transactions and data (such as WebSphere Extended Personalization Offering)

- User-to-Online Buying: commerce applications where users are interacting with e-commerce systems (such as WebSphere Commerce Suite)

- Business-to-Business: extended enterprise applications that integrate programmatic interactions between organizations (such as WebSphere B2B Integrator)

- User-to-Data: information aggregation applications where tools extract information from other data sources (such as WebSphere Host Integration and IBM Content Management)

- User-to-User: collaboration applications where tools facilitate communication between users (such as Lotus Collaborative Services)

- Application Integration: connecting disparate enterprise applications or systems (such as MQSeries family of products)

Business Partner expertise and solutions are beyond the scope of this book. As for a comprehensive portfolio of leadership products? That is what this book is about!

Framework Programming Model

Having a strategy is all well and good, but what about practical implementations? IBM's Framework for e-business is built on a set of four key principles that translate into supported standards and technologies. These four key principles are

- Maximize ease and speed of development and deployment. This means that the products found in the portfolio have been built or adapted to a server centric, Java based component model and toolset. This allows rapid collaborative development and deployment by those with skills ranging from non-technical graphic designers to programmers.

- Accommodate any client device. The Framework supports Internet standards and a server centric model that expands access to a broad range of client devices.

- Ensure portability across a diverse server environment. The Framework supports the open, unifying Java platform to make it easy to deploy applications on multiple types of operating systems and hardware. IBM's support of the Linux operating system on it's existing hardware and operating systems provides additional support to this principle.

- Leverage and extend existing assets. This principle asserts that e-business applications must extend the reach of existing applications that are the core of many businesses today, rather than always having to build from the ground up.

To support these principles, you need a Web application topology—a high-level blueprint, if you will, of the basic components of an application. At this point, we are still talking about a logical topology. Physical deployment of the different components may place all the components except the client on the same physical machine or put multiple copies of the components on multiple machines. The actual configuration depends on the application you are deploying.

There are four major components of the Web application topology that are components of the Framework's programming model. With these components, IBM has developed suggested design principles and designated supported technologies. The four components are:

- Clients: Clients can be either Internet or intranet enabled—they can be desktops, mobile laptops, cell phones, or embedded chips in your refrigerator. Their function is to accept and validate user input, communicate with Web application serv-

ers, and present results to the user. Built in the "thin" client model, this component contains minimal business logic or data.

- Web application servers: These servers receive requests, access business logic and data and return completed requests back to the client. They also provide the development environment for writing the business logic part of a Web application through the use of programming, data access, and application integration services. Finally, Web application servers provide a "runtime" environment for the e-business application.

- Infrastructure services: These services surround the Web application, providing the application server with load balancing, caching, directory and security services. One of the components of these services is the firewall. The firewall shields the company's network from outside interference from hackers or other unauthorized users.

- External services: These are existing, mission critical applications within the company, as well as access to the data and processes needed outside the firewall, such as information from pervasive devices and suppliers.

Clients

Clients are the primary interface with the user—although the user is not always human! They can be anything from the chip inside your "intelligent" refrigerator, through the smaller devices such as pagers, digital telephones and PDAs, to network and personal computers. In order to have communication with these devices that have different hardware configurations, certain standard Web-based technologies must be used. These include Java components, TCP/IP, HTTP, HTTPS, HTML, DHTML, XML, MIME, SMTP, IIOP, WML, and VoiceXML among others. These technologies are described later in this chapter.

There are two ways to develop a user interface from the client to the application server (Figure 1.2). The first is to use the HTML Client Model. In this model the request for information is made from

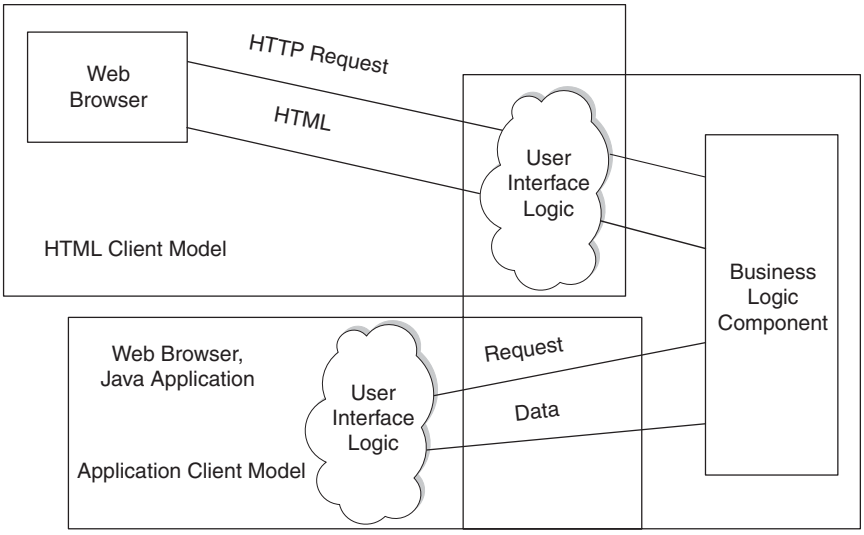


Figure 1.2. User interface models.

the client in the form of an HTTP request. The HTTP engine receives the request and gets the information that it needs from the business logic portion of the Web application server. HTTP (or another service, depending on the configuration) assembles the page and passes it back to the client. The advantage to this model is its extreme portability. Any Web browser can run a Web application based upon HTTP and HTML. In addition, the client part of the application is small and can download to the client quickly. Finally, the server can tailor the page depending on the client or the user requesting the information. The major disadvantage of this model is the difficulty of creating highly interactive user interfaces. Technologies such as dynamic HTML that use JavaScript are overcoming this difficulty.

The second model is to use the Application Client Model. In this model, the HTML page is generated and remains on the client. The client can then interact with the Web application server business logic via HTTP or IIOP. The only thing that comes back to the client is content. The client then interprets the data and displays it to the user. The primary advantage to this model is that it distributes the user interface generation to each client, reducing the load on the Web ap-

plication server. A second advantage is that this model is more familiar to application developers with a client/server background. The major disadvantage is that the client part of the application is small and takes more time to download and initialize. This restricts the clients to which the application can be deployed.

Web Application Server

The Web application server receives a request from the client user interface as described above. This may be a single process—show me all the white button-down shirts you have in your catalog. It can be an interactive process—allow me to purchase this shirt in a specific size. Regardless of how many times or what content is retrieved and displayed on the browser, the process is essentially the same. The Web application server receives the request, performs the business logic and retrieves the information required, and sends back the information and, possibly, a new Web page. There are several components (Figure 1.3) that can be used to build this process, known as

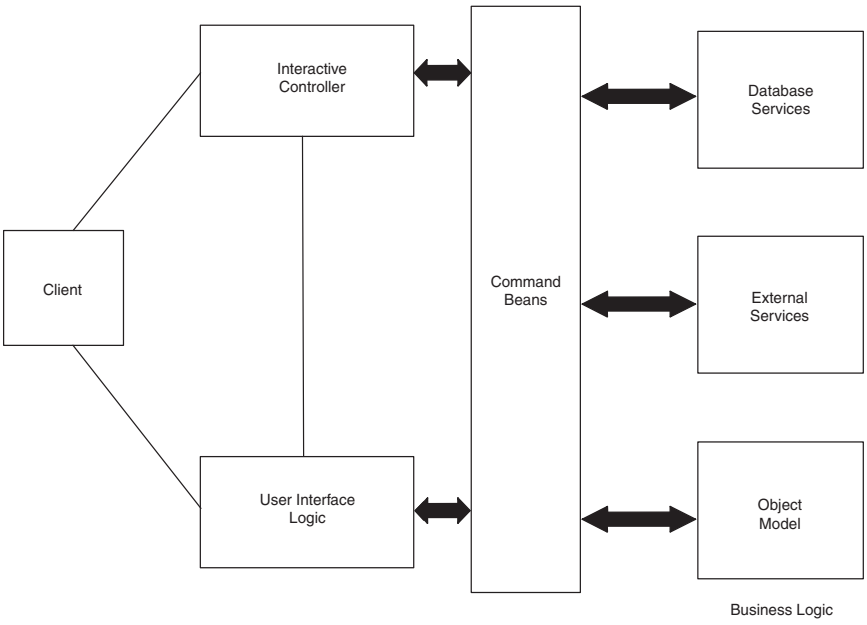


Figure 1.3. Web interactive model.

the Web interactive model. Remember, we are still discussing logical concepts at this point in time.

The Interactive Controller ties the client logic to a Web application. It is in charge of mapping the HTTP protocol specific input from the client request to the input required by the business logic residing on the server in order to fulfill that request. In doing this it can call services to verify security, perform business logic, access internal or external data, etc. When its task is completed, it enlists the UI logic component that creates the response page for the client.

The Framework supports the development of Interactive Controller logic using either Java servlets or Java Server Pages (JSP) technology. A Java servlet is an applet that runs on a server—the Java alternative to a CGI program. JSP technology allows scripting using Java or other scripting languages such as JavaScript, Basic, and REXX to be built into HTML response pages. This improves the ability of HTML pages to be dynamically created based on the content received. For example, if our shopper was looking for a Chinese-made silk white button-down shirt, the catalog might only return one shirt. JSPs allow the text on a page to be coordinated with the content. In this case the page would say, "Here is the shirt that you are looking for!" rather than "Here are the shirts that you are looking for!"

The User Interface Logic component was discussed under clients. It obtains the data either from the Interactive Controller or from its own access of the business logic. The Framework supports the development of User Interface Logic using servlets or JSP technology.

The business logic part of the Web application is separate from the details of Web technology. The request comes in, is handled by the User Interface Logic and Interactive Controllers, and then information is retrieved based on business rules and data. However, it is important to consider the interface between the User Interface Logic, Interactive Controllers, and the business logic. The Framework recommends that the business logic be wrapped with JavaBeans it calls command beans. JavaBeans provide a standard model for customizable components supported by a wide range of tools. This makes them part of the integrated end-to-end structure that Web businesses require. (A more detailed explanation of JavaBeans appears later in this chapter.) The Framework recommends that each command bean represent a single function.

The use of command beans allows for the isolation of changes. By keeping the execution of the bean separate from the logic it contains, the business logic can be changed as needed without affecting the interface. It is like changing CDs in your player—the interface is the same—a silicon disk read by a light—but the content is different. Command beans also allow for effective and efficient distribution of content. They can be serialized, and sent to a remote server, executed, and then sent back.

The Framework expands the JavaBean concept to use Enterprise JavaBeans (EJBs). EJBs also execute on an application server, but can be held inside a construct called a container. The container provides management and control functions, such as security, for the EJB. This further isolates the business logic from having to deal with any underlying technical changes. To go back to our music example, if we have a tape of the Beatles' Yellow Submarine and a CD of the same music, we can't use the same mechanism to play the music. But if we had a container that held all the bits of the music that could be wrapped in a plastic tape cartridge or enclosed in a CD, depending on where it needed to be played, that would serve a similar function to the container of EJBs.

There are two other Web application services that we should talk about: database services and collaboration services.

Database services integrate content with the Web application services. This content includes both operational data and multimedia content, as well as "e-analyze" content represented by online analytical processing (OLAP) applications. The Framework supports these services by use of Java access mechanisms such as JDBC and SQLJ to obtain the data for the Web application server. DB2 Universal Database, discussed in Chapter 5, supports the Framework's database services.

Collaboration services provide a shared virtual workspace to support business workflow processes. A highly developed collaborative process involves the use of information from any resource and access to that content by people who need to get to it. Workflow processes include the ability to route and track documents and to coordinate activities across your entire organization—and beyond. Lotus Domino Application Server, discussed in Chapter 5, supports the Framework's database services, as do several other architectural elements such as MQSeries Workflow, that work with Domino Application Server.

External Services

External services, also known as application integration services, address the need to access existing computing assets as well as the need to integrate existing business applications in order to provide a cohesive view of the business to a user. The Framework provides the following methods of application integration:

- Connectors are software components that provide linkage between the Web application server and the services it needs via application specific protocols. Connectors can be found in a wide range of existing data and application servers such as CICS, IMS, DB2, Domino, and MQSeries.

- Application Messaging services provide message-based communication between applications. MQSeries is IBM's primary Application Messaging service.

- Business process integration and workflow services extend application messaging services to include message brokering, intelligent message routing and message translation. Extensions to MQSeries are provided by MQSeries Workflow and MQSeries Integrator.

- Component integration services allow the wrapping of existing application logic to provide objects that the Web application can then access. Component integration services are provided by WebSphere Component Broker.

Infrastructure

When an application is running, there are certain services it needs—primarily security and insurance of availability. The Framework provides the following facilities to support Web applications:

- Directory support—A directory is required to keep track of users, groups, roles, and network policies to allow access to the application in a secure manner. The industry standard for

this application is an LDAP directory. IBM supports these directory services through the SecureWay Directory.

- Certification authority—Web security uses security certificates to make sure that they are talking to the person they think they are talking to. Public key infrastructure (PKI) technology is emerging as the preferred method to provide this service. A certification authority is needed to provide digital credentials and security features such as message integrity, data privacy, signature verification, and user authentication. IBM supports certification services through Tivoli SecureWay Public Key Infrastructure.

- Firewall support—As noted earlier a firewall protects your data from outside danger such as hackers and unauthorized users accessing your data. Firewalls can also be used to provide encryption services to send encrypted messages outside the wall. IBM supports these services through SecureWay Firewall.

- Proxy server—Proxy services serve as a conduit between the client and Web application server by forwarding client requests to the server and server answers back to the clients. This provides added security for clients that reach outside your firewall to access information. IBM supports proxy services through WebSphere Edge Server.

- Network Dispatcher—A network dispatcher aids in load balancing. The dispatcher serves as one network address for access to the server. The proxy dispatcher can then route the request to different physical servers acting as one logical server. IBM supports network dispatcher services through WebSphere Edge Server.

With these four major components: clients, Web application server, external services, and infrastructure, an effective Web application can be built. IBM has provided tools in every component category. As of this writing, there is movement within the Software Group to consolidate and combine these tools to insure that all aspects of e-business can be built with IBM tools. At the same time, by following standards, IBM insures that these tools and products are compatible with other vendors' tools and products that follow standards.

WebSphere Software Platform

In June 2000, IBM announced the WebSphere Software Platform as an implementation of the Framework for e-business. This platform brings IBM to tighter tools integration and increased consolidation of products into solutions that match companies' business needs. Built on the WebSphere brand, the platform strategy divides the IBM software products into four main groups: Foundation, Foundation Extensions, Application Accelerators, and Customer and Partner Applications.

The Foundation layer is focused at the basic customer need that all e-businesses have—delivering a business process using Web technology, quickly! These business processes are the same types of products that are found in the Framework for e-business logical "build" component of the Web application server. The Foundation provides the basic tools to handle the e-business transactions. It also provides the ability to extend the Web application to the back-end systems, thus enabling these applications to the Web as described in the external services portion of the Framework for e-business. Foundation business solutions include WebSphere Application Servers to store information and provide the run-time environment and application integration solutions to move data from one program to another and one application to another and extend back-end applications to the Web.

The Foundation Extensions are also part of the logical "build" component of the Web application server. The extensions provide integrated services and tools that can be used as companies grow their e-business. The extensions are divided into Development, Presentation, and Deployment groups. By picking and choosing from these services and tools, a company can rapidly develop and deploy Web applications.

The Development tools comprise products needed to develop Web applications, such as VisualAge for Java. The Presentation group includes products that make content available to the Internet, such as WebSphere Portal Server. This group includes both browser presentation of Internet content, as well as presentation of content through pervasive devices. The offerings extend to a presentation of a customized user experience. The customized experience includes a portal, content management, personalization, site analysis, and a solid Web application server foundation.

The final group, Deployment, includes tools that make the business application available to a variety of client devices, including pervasive devices, managing deployment to servers and assessment of existing applications. These offerings can be complimented with Tivoli Policy Director. Mobile computing and one-to-one personalized marketing require extensive security requirements including role and rule-based authentication.

Application Accelerators are a combination of the technology from Foundation and Foundation Services aimed at a specific application needs plus components, templates, and tooling, adding value to help the customer quickly differentiate their application and business models for four main types of e-business applications: e-commerce, workflow, collaboration, and business-to-business (B2B) integration. For example, with WebSphere Commerce Suite, a customer can quickly develop a store-front application on the web. This suite comes with components able to manage the catalog, payment, security, etc. required for an Internet store-front application. They provide enough tools for a company to begin development of one of these types of applications, yet retain the standards approach to which IBM is committed. As we explore IBM's solutions further, you will see how they can all be used interchangeably to grow applications as companies need.

At the highest level, Customer and Partner Applications are ready to implement business services. WebSphere provides Ready-to-Run support for these types of applications, predominantly built by about 20,000 IBM business partners and about 9,000 independent software vendors. These applications are geared for such e-business strategies as B2B, business to customer (B2C), business to employee (B2E), Process Automation and e-Markets. They also include types of applications like SAP (Enterprise Resource Planning) and Siebel (Customer Relationship Management) that have been externalized to the Web.

What does the WebSphere Software Platform do for you? Is it just a cafeteria selection of products? No, it is an effort to provide integrated offerings for new market categories, an expression of the IBM Framework for e-business. IBM made a decision a few years ago not to be an application company. They have left the application playing field to companies like Siebel and SAP. Their strategy is to provide the infrastructure IBM's business partners need for development and deployment of their product. WebSphere Software Platform is an

extension of the IBM Framework for e-business in that it provides companies the ability to select the tools, products and services they need to do their business in a manner that differentiates them from the competitors. The catch is that companies need to understand what they need before they select from the portfolio of tools and products. In the case of a company that does not have the skills or experience to make this selection, IBM Global Services can provide the expertise to work through a company's requirements and determine the selections it needs.

IBM has created the application accelerators to allow application providers to quickly take advantage of IBM's e-infrastructure. Packages such as WebSphere Commerce Suite and WebSphere B2B Integrator Suite are already available for specific types of e-business applications. In addition, IBM is working closely with many application builders such as Siebel to bring a firm foundation of infrastructure software to these companies' pre-built applications.

Concepts to Make an e-Business Work

The rest of this chapter is devoted to an exploration of e-business buzzwords, theories, and concepts. One of the great challenges of dealing with technology is keeping up with the changing theories and buzzwords and relating them to existing architectures. If you have spent any time in this business, you know that some new concepts bear a relationship to old, yet there is frequently a different twist. The trick is to find enough of the similarities to grasp the fundamentals. Let's look at some of these concepts.

In order to create an e-business that can satisfy customers and provide the company the benefits that it needs, there are several requirements that need to be satisfied: transaction persistence (the ability for the transaction to exist for more time than it takes to download it from the Internet), transaction security, scalability (the ability of an application to handle as many users as it needs to), content accuracy, and self-learning transactions (remembering the things that you have done previously in a transaction and being able to repeat them).

Persistence is a concept borrowed from the object world. It implies that an object exists in its current state after you are done manipulating the object. For example, even though I am no longer

connected to my online bank and I am done making entries to it, it will still exist in the state that I last left it—with money associated to the account. This becomes more critical when you are in the middle of an Internet transaction. There are some sites that don't remember much as you go through a transaction. For example, suppose you fill out a form describing in loving detail your Great-Aunt Harriet's crystal vase for an auction site and forget to fill in one of the required fields. You press Enter and the computer screen changes to an error message telling you that you did not fill in one of the required fields and to click the "back" button to finish the transaction. You follow instructions, click the back button only to find out that the form is blank and your lovingly created description is gone. Had your description remained, then you would be dealing with persistence.

Transaction security applies to everything about a transaction—the content as it exists everywhere—the company's data storage, your computer, and all lines in between. It insures that the credit card number you enter onto a Web form is encrypted before it leaves the safety of your computer. It also insures that no unauthorized person can look at the information you provide. It also implies a level of trust that you have in the entity with which you are doing business—that they will not give your information to another group without your consent. Privacy, an off-shoot concern of security, is a large concern for everyone.

Scalability, while not strictly an e-business concept, is highly important to e-business, particularly Web sites. When building an in-house system, there is some effort made to gather information about how many users will be on the system at a particular point in time. No such research is available for the Internet. So most Web site developers are of two minds—please let everyone find my site and please don't all come at once! But, of course, they do and if you are unlucky, your system slows or crashes and they never come back. Another site is only a click away.

Content accuracy is contentious. Some of it is due to our own ability to confuse things. Take nicknames for example. It is easy for a computer software program to discern that Bill is William, but more difficult to understand that Casey, Carol, and Carol Ann are the same person. More than once I have found myself getting two copies of the same magazine after renewal—only to find that one of them has a slight variation of address. A friend of mine separates the source of his junk mail by using variations of his name for the

different magazines to which he subscribes. For example, using D.P. Snow for his subscription to Forbes and Danny Snow for his subscription to Outdoor Life. This allows him to understand which magazine is selling its mailing lists. However, it doesn't help any company trying to understand customers by what they read. And then, of course, there is what the company does to mangle the information. One company found they had over a thousand customers with the same social security number. It seemed that when the data entry clerk could not find the social security number, she entered her own. This was bad enough when data was kept internally, but now that a user gets to see her data in all its living color, accuracy is becoming more of an issue.

Tag Languages

Tag languages communicate information about the data with which they are sent. They do this by associating tags with content to provide information about the content, either content meaning or how the content should be displayed. Keywords or characters are enclosed in angle brackets (<>) that are sent from one place to another with the content. Parsers at either end of the transaction interpret the keyword or character in order to display or understand the data.

SGML (Standardized General Markup Language) is one of the oldest tag languages. It never caught on with the Web community because it is very difficult to master. It has, however, formed the basis for today's most popular tag languages: HTML and XML.

HTML provides users with a simple, easy-to-use language for creating Web sites and linking to other Web pages for simplified Web navigation. Released in 1986, it is now in its fourth version. HTML is the most widely used vehicle for delivering information over the Web. Most Web pages you see are stored and transmitted in HTML and are easily accessed by a browser that contains an HTML parser.

HTML tells the browser how that item is to be displayed on the screen. In the following

```
<b>$5.98</b>
```

<b> indicates that the price "$5.98" should be displayed on the screen in bold. It does not explain whether $5.98 is the cost of the

item, the sale price of the item, or even if the price is in Canadian, U.S., or Australian dollars. Not only is that meaning unclear to humans, it is also unclear to search engines looking for the lowest price of a particular item. Because HTML only describes how content should be displayed, HTML documents are called unstructured documents.

This problem is compounded when companies try to transmit information to each other. Companies have been sending information to each other for many years using a format called Electronic Data Interchange (EDI). EDI is a powerful tool that has been used by a number of companies to exchange data. The companies agree to a pre-determined format with the appropriate headers and trailers giving number of records sent, date sent, etc. The receiving company could tell where the information they needed was in each record by its location and physical description (type, length, etc.). In addition to requiring extensive agreement to the format of the records, EDI transmissions could only be exchanged between companies with like hardware and operating systems. Company A could not send an EDI file created on OS/390 to Company B who wanted to receive it on a computer with an AIX operating system.

XML (Extended Markup Language) is a markup language that can be used to create structured documents that are stored and transmitted over the Web. It separates the content and what it means from how the content is to be displayed. This means that the only thing that any two companies need to agree on to transfer data is the set of XML tags that will be used with the transmitted content. XML is also extensible because it allows the creation of new tags without having to have the tags approved by a standards board.

An example of an XML tag is

```
<PRICE type = "sale" unit="US Dollar">$5.98</PRICE>
```

tells us that $5.98 is the sale price in U.S. dollars, not suggested retail price in Canadian dollars. An XML-enabled search engine can interpret this tag when looking for the lowest price of a particular item.

XML extensions have proliferated, many of them defined by industry or function, such as cXML for catalog content. These extended tags used by a particular document are stored in a file called the docu-

ment type definition (DTD). This file can reside on the same physical location as the XML document, or on a different server accessed by multiple applications. Thus, if a chemical company and its associated suppliers have agreed upon a set of XML extensions, the companies only need to have one DTD apiece to enable them to parse the XML documents.

XSL (EXtensible Style Language) is an XML language for reformatting and presentation, similar in concept to HTML. Style sheets in XSL can be associated with a document as it is displayed on different devices or in different formats. For example, a retailer could develop a catalog page that can be used in a hard-copy paper catalog and also displayed on a Web page. He can then associate two different XSL style sheets. In addition to flexibility for the media in which the sheet is displayed, the separation of style and content description allows the retailer the ability to change the price of an item in only one place—the original XML page—and have it readily available in multiple mediums.

IBM is developing and strongly supporting XML. In combination with TCP/IP as the universal connectivity protocol connecting everything from mainframes to your washing machine, Java as the "write once, run anywhere" programming language, XML extends the universality of e-business to content. To that end, IBM is supporting the following:

- DB2 XML Extender allows XML documents to be stored in DB2 as a collection of data items in multiple columns and tables.

- DB2 Text Extender supports search functions that can be applied to a section or a list of sections within a set of XML documents.

- Xeena is an XML editor that interfaces with a DTD to validate XML tags.

- XML Parser for Java allows users to generate new XML documents or manipulate existing documents.

- XML Parser for C++ provides the same ability in C++.

- Visual DTD is a visual tool for creating, viewing and editing DTDs.
- XSL Trace allows you to step through XSL scripts and shows you transformation rules, XML, and HTML as they are created.
- YODA (Your Own Data Access) allows the use of XML tags and XSL to create files for EDI exchange with organizations that require an EDI format, such as some U.S. government agencies.
- SOAP (Simple Object Access Protocol) is a communications technology and messaging mechanism based on XML.

These, and other XML-based products, can be found on IBM's alphaWorks Web site.

Java

Just as early bulletin boards, HTML, and the World Wide Web changed Internet computing, Java changed the ability of companies to take advantage of Web computing and transform their businesses into e-businesses. Prior to the Java computing model, computer programming was generally platform dependent. Different languages were developed for different operating systems (discussed further in Chapter Three). While there was some portability of languages to these operating systems, there was virtually no portability of a program from one platform to another. (In this context, platform means a combination of the operating system and the hardware on which it runs.) If a program was built and compiled on an S/390 mainframe, there was no way it was going to run on an IBM Personal Computer with a Windows operating system, much less on a cell phone.

Java changed all that. Java was developed by Sun Microsystems to enable companies to build software that can run on many different types of computers, as well as pervasive devices. How does this work? Java runs on a piece of software called a Java virtual machine. It is virtual because the machine does not really exist, it just appears to exist. All Java programs are written as if they were running on this platform. The platform is then built into Web browsers, cell phones,

and the like. This virtual machine is charged with translating the Java virtual machine platform instructions into instructions of the machine that the virtual machine is running on. So, if the Java program is running on an S/390, the instructions are translated to OS/390 operating system instructions. If it is running on Windows NT on an IBM Netfinity, the Java Virtual Machine translates the instructions into Windows NT operating system instructions, and so on. The idea is to "write once, run anywhere."

Java programs are called applets. An applet is a program that can be executed from another application, but not from an operating system. They are small in size, cross-platform compatible, and highly secure. An applet that runs on a Web application server is called a servlet. There is also a concept in Java called a Java Bean. What is a Java Bean?

A Java Bean is a piece of software code that can be used as a component in a Web application. A component is software code that performs a certain function, such as rotating an object on a Web page. Beans are developed with object oriented theory in mind (more on object theory in Chapter Two). To that end, they share several object characteristics: introspection, customization, events, properties, and persistence.

Because a bean allows introspection, builder tools and programatic interfaces can understand how a particular bean works. This allows the programmer to manipulate the bean as needed. To that end, a bean that rotates an object only needs to be told by the programmer that the object needs to rotate and to what degree. The programmer doesn't need to create the code to perform the rotation.

Customization allows a user to alter the appearance and behavior of a bean. In our rotating example, the bean can be changed from general rotation of an object to a rotation of an object 180 degrees. That means that every time the bean is used, the object with which it is used rotates 180 degrees, no more, no less. This is similar to the object concept of inheritance.

Events are like triggers in object theory. Beans can be recipients of events as well as have the ability to fire events. For our rotating bean, if a button on a Web page is pressed, the bean can automatically rotate an object 180 degrees. This rotation can fire an event that flips another object on a page that causes another object to stream across the page, etc. This is one way to get all those whirling objects that take Web pages so long to download.

Beans have properties. These properties are the hooks that allow the customization and manipulation of the bean. For example, the rotating bean has a property that dictates the number of degrees it is able to rotate an object. Prior to customization, the programmer supplied that number. After customization, it was set to 180 degrees.

Finally, persistence allows the bean that has been customized to have the state of the bean saved and restored. This allows the developer to use the bean as originally built, and, at the same time use the customized bean.

Summary

This chapter focuses on developing the concepts of e-business, setting the stage for the chapters that follow. We have looked at IBM's focus on e-business fundamentals: integration and infrastructure and their strategy for supplying the software for these fundamentals—the Framework for e-business. Chapter 2 covers software architectures that have led to the development of the Framework. Chapter 3 describes the operating system foundation for this Framework. Chapters 4, 5, and 6 explore the Build, Run, and Manage components of IBM's e-business Framework.

We also looked at an implementation of the IBM Framework for e-business, WebSphere Software Platform. This platform shows how many of IBM's software products can be used to develop a secure, reliable and scalable Web application. Finally, we looked at some of the concepts underlying the rapid development of e-business—the usefulness of markup languages such as XML and HTML and the importance of Java to development.

The next chapter explores several important concepts that influence how software architectures are built and data is stored, including the relational theory that was developed in IBM's research labs. There is also a brief section on Business Intelligence, concepts that underlie the ability of companies to move to a form of e-business known as the ability to e-analyze. Below are a few case studies to show how e-business concepts can be used within a company, as well as brief definitions of some of the terms we have covered in this chapter.

Case Studies

Pristine Appliance Company

The Pristine Appliance Company has over 40 different fulfillment and financial systems to run its business. In the last decade, they have found this cumbersome and costly.

Bucking current wisdom of organizational theory that decentralization is good, the company has consolidated its fulfillment and financial systems. They have "e-enabled" their workforce. They have put a supply chain portal in place to connect trading partners, sellers, distributors, and back office operations. The same portal is re-used for the front end so that individual consumers can use it to order small appliances and accessories.

Following the guidelines of the IBM Framework for e-business, Pristine Appliances built its portal with IBM WebSphere Application Server, WebSphere Commerce Suite, HTTP Server, VisualAge for Java, and Commerce Integrator with MQSeries.

"By going with a Web-based B2B model, we've made life easier for our distributors and ourselves," said Marsh Hopkins, chief architect of the company. "Before, it was cumbersome, costly and time-consuming to get our appliances to our major trading partners."

The most exciting aspect of Pristine Appliance Company's transformation is every homeowner's dream—intelligent appliances. Since these are still machines, the intelligence is small, but they have an umbilical cord to the Web in case their appliance life is too much to handle.

"We look forward to this new world," Hopkins added, "and expect to see the same level of return on investment we've seen so far. Already we're seeing order processing savings in excess of 80 percent. We expect our B2B site to obtain its ROI in eight months, and the customer site in five months. We are very pleased with IBM software."

Pegasus Airline

Pegasus Airline had poor a customer satisfaction rating. Because of that, they were losing customers to competing airlines on the same

routes. They needed to establish themselves as a leader in providing value-added customer service solutions. They saw that rapid advances in wireless technology, such as cellular telephones and personal digital assistants (PDAs), might help them improve their customer service perception. They took their idea to IBM.

"We selected IBM Global Services because of the strength of their Pervasive Computing Group," said Tom Sure, CIO of Pegasus Airline. "We wanted to be working with a company that was a leader in the area of wireless computing and communications. In addition, IBM really understands our business and that's important to us."

IBM Global Services Pervasive Computing Group and Pegasus technology deployed an IBM Wireless Web-based Customer Information System in approximately two months. Pegasus Wireless delivers access to personal travel itineraries, flight arrival/departure and gate information, and Pegasus flight schedules from the Pegasus Web site to a variety of devices, including Web-enabled wireless telephones, and the PalmVII™ hand-held organizer. The solution used VisualAge for Java for development and relied heavily on Java servlet and XML technology.

"We're very pleased," said Sure. "We just received our quarterly customer service ratings from the Consumer Trade Group and they have improved 50 percent. We're well on our way to being regarded as the leader we know we are."

Definitions

- Applet is a program that can be executed from another application, but not from an operating system. They are small in size, cross-platform compatible, and highly secure. Web browsers that are equipped with Java virtual machines can interpret applets they obtain from Web servers.

- B2B (Business-to-Business) is the selling of goods and services between businesses.

- B2C (Business-to-Customer) is the selling of goods and services from a business to a customer.

- B2E (Business-to-Employee) is communication between a business and its employees.

- CGI (Common Gateway Interface) is a server-side specification that allows the transfer of information between a Web server and a program that conforms to the CGI specification.

- CORBA (Common Object Request Broker Architecture) is an architecture developed by OMG that is based on object-oriented theories.

- CRM (Customer Relationship Management) is a set of applications that helps businesses manage its customer relationships by tracking customers' histories and managing the customer in a coordinated way through all the company's sales channels.

- DHTML (Dynamic HTML) can be a page of Web content that changes each time it is viewed, depending on items such as time of day and the profile of the reader. Alternatively, it can be new HTML extensions that allow a Web page to react to user input without contacting a server.

- DTD (Document Type Definition) is a file that contains definitions of valid XML tags used in a particular document or set of documents.

- EDI (Electronic Data Interchange) is a tool to exchange data and support transactions from like platform to like platform.

- ERP (Enterprise Resource Planning) is a set of applications that automates the business of businesses—predominantly in areas such as finance, human resources, order processing and production scheduling.

- Extranet is an intranet that can be accessible to trusted users outside the firewall. Extranets are common means for business partners to exchange information.

- Firewall is a piece of software that shields an organization's network from exposure to hackers and other unauthorized users via the Internet.

- HTML (Hyper Text Markup Language) is a language associated with Web page content that tells the browser how that item is to be displayed on the screen.

- HTTP (HyperText Transfer Protocol) is the underlying World Wide Web protocol. This protocol defines how messages are formatted and transmitted as well as what actions Web servers and browsers should take in response to various commands. Each command is regarded as a separate, distinct command—nothing can be "remembered" from one command to the next. HTTP is called a stateless protocol. In response to this, scripting languages such as JavaScript were developed to enable Web authors to design interactive sites. JavaScript, like HTTP, can interact with HTML source code.

- HTTPS (HyperText Transfer Protocol Secure) is an HTTP site that requires SSL protocol.

- IIOP (Internet Inter-ORB Protocol) is a protocol developed by the Object Management Group (OMG) to implement CORBA solutions over the World Wide Web. IIOP's advantage over HTTP is that more complex objects such as binary large objects (BLOBs—often used for pictures) can be transmitted over the Internet. HTTP can only transmit text.

- Internet is a global network connecting millions of independent computers.

- Java is an object-oriented language developed by Sun Microsystems that can run on most computers using a Java Virtual Machine.

- JavaBean is a piece of software code that can be used as a component in a Web application.

- Java Virtual Machine is a Java interpreter and runtime environment that can run on most operating systems.

- JSP (Java Server Pages) is an HTML extension for doing server-side scripting in Web pages. They have special tags for inserting Java logic directly into the page.

- LDAP (Lightweight Directory Access Protocol) is a set of protocols for accessing directories that commonly contain information such as email addresses and security keys.

- MIME (Multipurpose Internet Mail Extensions) is a specification that formats content such as graphics, audio and video files so they can be sent over the Internet.

- OMG (Object Management Group) is a group of about 700 companies whose goal is to provide standards for developing object-oriented applications.

- PDA (Personal Digital Assistant) is a handheld device that can be used for computing, as a telephone and fax sender or receiver that contains the ability to be part of a network.

- SCM (Supply Chain Management) is a set of applications that helps businesses manage their supply chain, including managing warehouses, tracking inventory and distributing goods.

- Servlet is an applet that runs on a server—the Java alternative to a CGI program.

- SGML (Standardized General Markup Language) is one of the oldest tag languages and is the forerunner to HTML and XML.

- SMTP (Simple Mail Transfer Protocol) is a protocol that is used to send e-mail messages from one server to another or a client to a server.

- TCP/IP (Transmission Control Protocol/Internet Protocol) is a connectivity protocol that allows dissimilar devices to communicate with each other.

- WWW (World Wide Web) is a group of networked servers (Internet) that supports specially formatted documents.

- XML (eXtended Markup Language) is a standardized way to define the meaning of a piece of content.

- XSL (eXtended Style Language) is an XML language for reformatting and presentation.

- YODA (Your Own Data Access) is a Java applet that allows the use of XML and XSL to create EDI files.