

e-Commerce Patterns Using WebSphere Commerce Suite

Patterns for e-business Series

Selecting the topology and products for
your e-commerce solution

Guidelines for performance,
design, and development

Creating e-commerce sites
by example using Java



John Ganci
Paul Chik
Hernan Cunico
Al Hamid
Peter Kovari

ibm.com/redbooks

Redbooks



International Technical Support Organization

**e-Commerce Patterns
Using WebSphere Commerce Suite
Patterns for e-business Series**

October 2000

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix C, "Special notices" on page 451.

First Edition (October 2000)

This edition applies to Version 4.1 of IBM WebSphere Commerce Suite Pro Edition, for use with Windows NT and AIX operating systems.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HZ8 Building 678
P.O. Box 12195
Research Triangle Park, NC 27709-2195

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 2000. All rights reserved.
Note to U.S Government Users – Documentation related to restricted rights – Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

Preface	xiii
The team that wrote this redbook	xiii
Comments welcome	xv
<hr/>	
Part 1. User-to-Online Buying Pattern	1
Chapter 1. Introduction to Patterns	3
1.1 Patterns for e-business	3
1.2 User-to-Online Buying Pattern	5
1.3 How to use the patterns	6
1.4 Patterns for e-business Web site	6
1.5 Application Framework for e-business	7
1.6 An integrated view of e-business solutions	10
1.7 Structure of this User-to-Online Buying Pattern redbook	11
Chapter 2. Choosing the application topology	13
2.1 Application topology 1	14
2.1.1 Application topology 1: business driver	14
2.1.2 Application topology 1: key features	15
2.1.3 Application topology 1: considerations	15
2.1.4 Application topology 1: example	15
2.2 Application topology 2	15
2.2.1 Application topology 2: business driver	16
2.2.2 Application topology 2: key features	17
2.2.3 Application topology 2: considerations	18
2.2.4 Application topology 2: example	19
Chapter 3. Choosing the runtime topology	21
3.1 Node types overview	22
3.1.1 Web application server node	22
3.1.2 Public Key Infrastructure (PKI) node	23
3.1.3 Domain Name System (DNS) node	23
3.1.4 User node	23
3.1.5 Directory and security services node	23
3.1.6 Database server node	23
3.1.7 Protocol firewall and domain firewall nodes	24
3.1.8 Dispatcher node	24
3.1.9 Shared file system node	24
3.1.10 Web server redirector node	24
3.1.11 Application server node	25
3.1.12 Commerce server node	25

3.1.13 Application and data nodes	25
3.2 Runtime topology A	25
3.2.1 Runtime topology A: Example	29
3.3 Runtime topology B	32
3.3.1 Runtime topology B: Example	35
Chapter 4. Product mapping	39
4.1 Runtime topology A: product mapping	40
4.1.1 Runtime topology A: product mapping - Windows NT platform	40
4.1.2 Runtime topology A: product mapping - AIX platform	42
4.2 Runtime topology B: product mapping	43
4.2.1 Runtime topology B: product mapping - Windows NT platform	43
4.2.2 Runtime topology B: product mapping - AIX platform	46
<hr/>	
Part 2. User-to-Online Buying Pattern: guidelines	51
Chapter 5. Performance guidelines	53
5.1 Architecture and hardware capacity planning	53
5.1.1 1-tier runtime topology (single node)	53
5.1.2 2-tier runtime topology (single WCS server - remote DB server)	54
5.1.3 3-tier runtime topology (multiple WCS server - remote DB server)	54
5.1.4 Hardware capacity planning	54
5.2 Operating system and network	54
5.3 Web application servers and technologies	56
5.3.1 WebSphere Commerce Suite	56
5.3.2 Database server	66
5.3.3 Network Dispatcher	68
5.3.4 Integration to back-end system	70
5.4 Application development	72
5.4.1 Net.Data	72
5.4.2 HTML pages	73
5.4.3 Servlets and JSPs	73
5.5 General performance guidelines	76
5.6 Where to find more information	76
Chapter 6. Technology options	77
6.1 Web clients	77
6.1.1 Web browser	78
6.1.2 Markup languages	79
6.1.3 Client-side scripts	82
6.1.4 Java applets	82
6.2 WebSphere Application Server	84
6.2.1 XML	85

6.2.2	JavaServer Page (JSP)	86
6.2.3	Java Servlets	86
6.2.4	JavaBeans	87
6.2.5	Enterprise JavaBeans (EJB)	87
6.3	WebSphere Commerce Suite	89
6.3.1	Commands	91
6.3.2	Tasks	92
6.3.3	Overridable functions	92
6.3.4	Database	94
6.3.5	CGI implementation	94
6.3.6	WebSphere Application Server integration	98
6.3.7	Connectors	101
6.3.8	Mass import XML	102
6.3.9	Additional enterprise Java APIs	103
6.4	Where to find more information	104
Chapter 7. Application design guidelines		105
7.1	General Web application design guidelines	105
7.2	Application elements	106
7.2.1	Web clients	108
7.2.2	WebSphere Commerce Suite	108
7.2.3	Back-end systems	111
7.3	Application structure	111
7.4	Working logic	112
7.4.1	Net.Data, C++ modules	112
7.4.2	JavaServer Pages, Java Servlets and JavaBeans	113
7.4.3	Business logic	114
7.4.4	Presentation logic	116
7.4.5	JavaServer Pages or Net.Data macros	118
7.5	Controller	119
7.6	Page construction	119
7.7	User tracking and session management	120
7.7.1	Cookies and URL rewriting	120
7.7.2	Session persistence and clustering	121
7.8	Response time	122
7.9	Security	123
7.9.1	Authentication	123
7.9.2	User Registry	124
7.9.3	Access control	124
7.9.4	Integrity	126
7.9.5	Cross-referencing	126
7.9.6	WebSphere Application Server	127
7.10	e-commerce model	127

7.10.1 Model overview	127
7.10.2 Use case example	130
7.10.3 Technical walkthrough example	138
7.11 Where to find more information	151
Chapter 8. Application development guidelines	153
8.1 Development process overview	153
8.1.1 Application and architecture domains	155
8.2 Solution outline	155
8.3 Macro design	156
8.4 Micro design	158
8.5 Build cycle	159
8.5.1 Develop source code	160
8.6 Deployment	163
8.6.1 Deployment: work products	163
8.6.2 Deployment: process	164
8.6.3 Deployment: tools	164
8.7 Organization role	165
8.8 e-Commerce application functionality	166
8.9 Development environment	168
8.9.1 IBM Application Framework for e-business	168
8.9.2 Development tools	170
8.9.3 WebSphere Commerce Suite Administration	178
8.10 Commands, tasks, overridable functions	183
8.10.1 Commands	183
8.10.2 Tasks	184
8.10.3 Overridable functions	184
8.11 Programming logic behind the application	185
8.11.1 Business logic	185
8.11.2 Presentation logic	185
8.12 Where to find more information	185
Chapter 9. System management and security guidelines	187
9.1 Systems management guidelines	187
9.1.1 Managing your WebSphere Commerce Site	188
9.1.2 WebSphere resource management	189
9.2 Systems management product guidelines	190
9.2.1 WebSphere Application Server Administrator's Console	190
9.2.2 Site Analyzer	197
9.2.3 WebSphere Commerce Suite	199
9.2.4 SecureWay Directory Management	203
9.2.5 HTTP Server Management	203
9.2.6 DB2 UDB Management	203

9.3 Security guidelines	207
9.3.1 Physical systems security	207
9.3.2 Operating systems security	208
9.3.3 Network security	209
9.3.4 Web application security	210
9.3.5 WebSphere security model and policy	213
9.3.6 HTTP single sign-on (SSO)	215
9.4 Backup and recovery guidelines	215
9.4.1 Using Tivoli Storage Manager (TSM)	216
9.4.2 Application backup and recovery	220
9.4.3 Guidelines for backup and recovery	222
9.5 Where to find more information	223
<hr/>	
Part 3. A working example	225
Chapter 10. Working example overview	227
Chapter 11. Runtime environment - AIX platform	229
11.1 Remote DB2 database server installation	230
11.1.1 Pre-installation requirements	231
11.1.2 Pre-installation steps	231
11.1.3 Install IBM DB2 UDB EE V6.1.0.6 for AIX	235
11.2 WebSphere Commerce server installation	237
11.2.1 Pre-Installation requirements	237
11.2.2 Pre-installation steps	238
11.2.3 Install IBM JDK V1.1.8 for AIX	241
11.2.4 Install IBM HTTP Server (Apache) for AIX V1.3.6.2	244
11.2.5 Install DB2 Client	251
11.2.6 Install WebSphere Application Server	252
11.2.7 Installing IBM Net.Data for AIX V6.1	259
11.2.8 Installing WebSphere Commerce Suite V4.1	260
11.3 WebSphere Commerce Suite configuration	262
11.3.1 Pre-configuration tasks	262
11.3.2 Create the WebSphere Commerce Suite Instance	264
11.3.3 Post-configuration steps	267
Chapter 12. Development environment	269
12.1 WebSphere Commerce Suite test environment	269
12.2 WebSphere Commerce Studio	269
12.2.1 WebSphere Commerce Studio overview	270
12.2.2 WebSphere Commerce Studio installation	271
12.2.3 VisualAge for Java configuration	272
12.3 C++ CGI development	272

Chapter 13. Create and publish a store	273
13.1 Create a store using Store Creator	273
13.2 Publishing a WebSphere Commerce Studio project	275
13.2.1 Defining the target publishing host	275
13.2.2 Configuring the target publishing host - AIX platform	277
13.2.3 Configuring the WebSphere Commerce server	280
13.2.4 Verifying the base store	282
13.3 Customizing a store	282
13.3.1 Add Category and Product data - massimport XML	286
13.3.2 Verify the example store	287
Chapter 14. SecureWay Directory (LDAP) - AIX platform	289
14.1 Overview	289
14.1.1 What is a directory?	289
14.1.2 What is LDAP?	290
14.1.3 Why should I use LDAP?	290
14.1.4 IBM SecureWay Directory and WebSphere Commerce Suite	291
14.2 SecureWay Directory installation	291
14.2.1 Pre-installation requirements	291
14.2.2 IBM SecureWay Directory V3.1.1.2 for AIX installation	294
14.2.3 Install verification	295
14.3 SecureWay Directory configuration	296
14.3.1 Configure SecureWay Directory using a GUI	296
14.4 SecureWay Directory Server Administration	301
14.4.1 Start and stop the server	301
14.4.2 Add and delete suffixes	302
14.4.3 Add entries to the directory database	304
14.5 SecureWay Directory integration with WebSphere Commerce Suite	308
14.5.1 Configure WebSphere Commerce instance	310
14.5.2 Verify the configuration	312
14.6 SecureWay Directory SSL configuration	314
14.6.1 Prerequisites	315
14.6.2 SecureWay Directory SSL configuration	315
14.6.3 Security Verification	316
14.7 SecureWay Directory Management Tool (DMT)	317
14.7.1 Connecting to directory servers	318
14.7.2 Display server properties	319
14.7.3 Administering schema object classes and attributes	319
14.7.4 Administering a directory tree	322
14.7.5 Administering directory entry ACLs	325
14.7.6 Troubleshooting	326
14.8 Where to find more information	326

Chapter 15. Java technologies in WebSphere Commerce Suite	327
15.1 Overview of Java technologies in WebSphere Commerce Suite . . .	327
15.1.1 JavaServer Page (JSP)	327
15.1.2 Java Servlets	332
15.1.3 JavaBeans	334
15.2 Configuring the WebSphere Application Server	336
15.3 JSP and bean example	337
15.3.1 Creating a JSP category list page template	337
15.3.2 Creating a JSP product list page template	344
15.3.3 Creating a JSP product page template	347
15.3.4 Testing JSPs	356
15.3.5 Deploying JSPs	357
15.3.6 JSP development recommendations.	361
15.4 Custom servlet and bean example	362
15.4.1 Creating a database query	364
15.4.2 Creating a bean and servlet	366
15.4.3 Modify the existing product display JSP	367
15.4.4 Publish the files under WebSphere Commerce Studio	370
15.4.5 Deploy the servlet under WebSphere Commerce Suite.	370
15.4.6 Testing the new servlet and bean	372
15.4.7 Integrating into the store.	373
15.4.8 Testing pages under the store	377
15.5 Where to find more information	378
Chapter 16. Personalization in WebSphere Commerce Suite	379
16.1 Personalization overview	379
16.1.1 Blaze Advisor Builder	380
16.2 Personalization example	382
16.2.1 Personalization worksheets	383
16.2.2 Test the store before personalization changes	384
16.2.3 Create the rule project and ruleflow	384
16.2.4 Create the ruleset and assign task	386
16.2.5 Compile the rule project	389
16.2.6 Modify the net.data macro associated view task	390
16.2.7 Publish the project	392
16.2.8 WebSphere Commerce Suite personalization configuration	393
16.2.9 Test the store after the personalization changes.	398
16.2.10 Maintain the rule project	398
16.3 Personalization debug tips	399
Chapter 17. Back-end integration using MQSeries XML messages	401
17.1 WebSphere Commerce Suite integration with MQSeries	401
17.1.1 MQSeries inbound XML	402

17.1.2 MQSeries outbound XML	402
17.2 MQSeries client/server runtime environment	403
17.2.1 MQSeries server V5.1 for AIX.	403
17.2.2 MQSeries client V5.1 for AIX	408
17.3 MQSeries adapter	410
17.3.1 Pre-configuration of MQSeries adapter.	410
17.3.2 Enable MQSeries adapter.	411
17.3.3 Access control for the MQSeries adapter	411
17.3.4 Enable MQSeries XML messages.	412
17.4 MQSeries XML message example	413
17.4.1 WebSphere Commerce inventory level checking example . . .	413
17.5 MQSeries adapter debugging tips	419
17.6 Where to find more information	421
Appendix A. Runtime environment - Windows NT platform	423
A.1 Remote DB2 database server installation.	423
A.1.1 Pre-installation requirements	423
A.1.2 IBM DB2 UDB EE V6.1.0.6 installation.	424
A.2 WebSphere Commerce Suite installation.	425
A.2.1 Pre-installation requirements	425
A.2.2 IBM JDK 1.1.7.	426
A.2.3 Install IBM HTTP Server V1.3.6.2 (Apache)	427
A.2.4 Install IBM DB2 UDB EE V6.1.0.6	432
A.2.5 Install WebSphere Application Server	435
A.2.6 Install IBM Net.Data V6.1	440
A.2.7 Install WebSphere Commerce Suite.	441
A.3 WebSphere Commerce Suite configuration	443
A.3.1 Configuration Pre-requisites	444
A.3.2 Configuring WebSphere Commerce Suite components.	444
Appendix B. SecureWay Directory (LDAP) - Windows NT platform . .	449
Appendix C. Special notices	451
Appendix D. Related publications	455
D.1 IBM Redbooks	455
D.2 IBM Redbooks collections	456
D.3 Other resources	456
D.4 Referenced Web sites	458
How to get IBM Redbooks	461
IBM Redbooks fax order form	462

Index	463
IBM Redbooks review	471

Preface

Patterns for e-business are a group of proven, reusable assets that can be used to increase the speed of developing and deploying Web applications. The pattern discussed in this book, User-to-Online Buying, identifies the interaction of users with enterprise Web sites that sell goods and services.

Part 1 of the redbook guides you through the process of selecting an application and runtime topology. The pattern provides you with platform-specific product mappings for implementing the chosen runtime topology.

Part 2 of the redbook provides a set of guidelines for building your e-commerce application. These guidelines include performance, technology options, application design, application development, systems management and security.

Part 3 of the redbook teaches you by example how to implement an e-commerce application using application topology 2. Some of the technology highlights include JSPs, servlets, beans, MQSeries XML messages, personalization, and LDAP.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, Raleigh Center.

John Ganci is a Senior Software Engineer for the IBM ITSO Raleigh Center. He writes extensively and teaches classes on WebSphere and related topics. John has 12 years of experience in product and application development, test and systems integration. Before joining the ITSO, he worked for the IBM Personal Systems Group, USA, developing e-commerce sites for IBM.

Paul Chik is an I/T Specialist on the e-business service team in IBM Hong Kong. Previously, he was a hardware engineer on AS/400. He has been working on e-business solution implementations for two years. He holds a degree in Physics with honors from the University of Hong Kong. His areas of expertise include e-business solutions, WebSphere Commerce, Lotus Domino, and WebSphere.

Hernan Cunico is an I/T Specialist in IBM Argentina. He has five years of experience in system administration and e-business field. He has worked at

IBM for three years. His areas of expertise include networking, Internet security, e-commerce, and e-business solutions.

Al Hamid is an I/T Architect in IBM USA. He has over 15 years of experience in several areas of computing including architecture, design and development of distributed computing systems, component and object-based application development, systems modeling and simulation, e-commerce, and Artificial Intelligence systems. He currently works for the IBM e-business National Practice.

Peter Kovari is a solution specialist on e-commerce in IBM Hungary. He has three years of experience on Web application development. Peter holds a Master of Science degree from the Technical University of Budapest. His current area of expertise includes e-commerce solutions.



Figure 1. The ITSO redbook Commerce Team: John, Peter, Paul, Hernan

Thanks to the following people for their invaluable contributions to this project:

John Rothwell is a Distinguished Engineer with IBM Global Services in IBM Canada. He is the lead architect for the Enterprise Solutions Structure (ESS) Reference Architecture for Online Buying, which this pattern is based on. ESS reference architectures are used by IBM Global Services when working on projects for their clients.

Jonathan Adams is an I/T consultant with IBM's Software Group and lead architect of the Patterns for e-business initiative. He works closely with all areas of IBM and industry consultants. His commitment to the idea of a systematic

approach to end-to-end e-business architecture is based on his many years of work in the field with major IBM customers in the United Kingdom.

Bill J Moore, IBM ITSO Raleigh Center

Carla Sadtler, IBM ITSO Raleigh Center

Geert Van De Putte, IBM ITSO Raleigh Center

Ken Ueno, IBM ITSO Raleigh Center

Margaret Ticknor, IBM ITSO Raleigh Center

Linda Robinson, IBM ITSO Raleigh Center

Gail Christensen, IBM ITSO Raleigh Center

Roland Doemer, IBM ITSO San Jose Center

Diane Pearce, IBM Canada

Norm Smith, IBM Raleigh

Mike Medicke, IBM Raleigh

Pete Smith, IBM Austin

Comments welcome

Your comments are important to us!

We want our redbooks to be as helpful as possible. Please send us your comments about this or other redbooks in one of the following ways:

- Fax the evaluation form found in “IBM Redbooks review” on page 471 to the fax number shown on the form.
- Use the online evaluation form found at ibm.com/redbooks
- Send your comments in an Internet note to redbook@us.ibm.com

Part 1. User-to-Online Buying Pattern

Chapter 1. Introduction to Patterns

We are all familiar with the pace of development of the computer industry during its relatively brief history. The rapid advances in computer hardware have been driven by the use of standards and well-specified components for assembly. The desire to apply these same approaches to software construction gave rise to object-oriented software, design patterns, and component-based development.

The concept of software design patterns was first published in *Design Patterns - Elements of Reusable Object-Oriented Software*. The software design patterns were inspired by the idea of patterns in the design of buildings, published in *A Pattern Language*. In the software industry, design patterns have gained acceptance by software architects and developers. Patterns enable an efficiency in the communication and implementation of software design.

The pattern concept has been applied to the systems architecture in, *Pattern-Oriented Software Architecture - A System of Patterns*. This book identifies patterns for systems architectures at a higher level than the original design patterns. The patterns are related to the macro-design of system components such as operating systems or network stacks.

I/T architects, encouraged by the success of design patterns, and facing challenges in systematic and repeatable description of systems, have also explored the idea of architectural patterns. IBM has made great advances in this area, in large part by the work done in the Enterprise Solution Structure (ESS) Reference Architecture. ESS looks at patterns from a complete end-to-end system architecture point of view. ESS promotes the assembly of proven designs and components from an ever growing and evolving repository, instead of hand crafting each solution from scratch. ESS is part of the IBM Global Services methodology. For more information on ESS see "Enterprise Solutions Structure", in the *IBM Systems Journal*, Volume 38, No. 1, 1999, found at: <http://www.research.ibm.com/journal/sj38-1.html/>.

1.1 Patterns for e-business

The Patterns for e-business aim to communicate in a highly accessible fashion the business pattern, systems architecture (application and runtime topologies), product mappings, and guidelines required for different classes of applications. The Patterns for e-business are cataloged according to the following business context schemes:

- User-to-Business
- User-to-Online Buying
- Business-to-Business
- User-to-Data
- User-to-User
- Application Integration

We anticipate that architects and designers will want to combine these patterns to compose solutions to more complex system architectures. For detailed information, refer to the Patterns for e-business Web site at: <http://www.ibm.com/software/developer/web/patterns/>.

Benefits of Patterns for e-business

The benefits of Patterns for e-business are increasingly gained from projects being:

- More profitable
By re-using existing components the resources the project cost will be reduced and you will be able to deploy your solution sooner.
- More complete
A more complete solution can be developed from the “pieces” contained within the patterns repository.
- Proven solutions
A large proportion of the end-to-end solution will have already been proved in a real environment.
- Integration
Standard interfaces and entry points exist to other IBM and IBM Business Partner products and services.

Who benefits from using Patterns for e-business?

The patterns can be used by:

- Pre-sales consultants
The patterns provide a base context for customer engagement discussions and presentations.
- Infrastructure consultants
The patterns provide information on the runtime topologies and product mappings, which can be used as a basis for an enterprise architecture.

- Application consultants

The patterns provide application design and development guidelines, which can be used for application solution development.

1.2 User-to-Online Buying Pattern

The User-to-Online Buying Pattern is a subset of the User-to-Business pattern, where products are sold through a catalog using such components as a shopping cart and/or wallet. The pattern can also include links to back-end systems allowing for inventory updates, order processing, delivery systems, and credit checking.

Examples of User-to-Online Buying include the following:

- Online retail industry
- Consumers purchasing goods
- Buyers purchasing goods from a supplier
- Company employees purchasing goods on behalf of their customers; for example, consumers ordering goods through a call center employee who is using an online buying system.

Electronic commerce (e-commerce) is the set of products and processes facilitating the secure purchase of goods and services over the Web, including such functions as:

- Advertising
- Marketing
- Shopping
- Purchasing
- Paying
- Shipping/delivery

e-Commerce solutions allow enterprises to reach new customers and manage transactions electronically. Consumers can purchase in confidence knowing their transactions are secure and their privacy is protected.

The material for this patterns was not created in an ivory tower. A high-level IBM task force examined several major online buying sites, and selected a dozen of the most successful e-commerce projects. For each project, they interviewed the lead architects, collected related information, and

documented the solutions in the Book of Case Studies for e-commerce Engagements.

In addition to documenting the individual projects, the authors developed a reference architecture for online buying following the approach of the Enterprise Solution Structure (ESS) and using terminology based on IBM's architecture description standard. This terminology is being introduced to the industry in general. The Book of Case Studies for e-commerce Engagements assists IBM Global Services practitioners in e-commerce projects by identifying the best practices and propagating their use across IBM. While the Book of Case Studies for e-commerce Engagements is an IBM confidential document, the User-to-Online Buying Pattern shares some of the authors' information with you.

1.3 How to use the patterns

The Patterns for e-business are particularly focused upon addressing common business application problems and providing answers to frequent architecture, design and implementation questions.

We recommend that you use the Patterns for e-business in a number of ways according to your needs:

- As a starting point for an end-to-end system architecture.
- As a detailed example and prescriptive approach, following the product mappings and guidance provided.
- As a way to design more complex, multi-channel systems, when several patterns are used together.

For maximum benefit, this redbook should be used in conjunction with experimenting with the WebSphere Commerce Suite V4.1 on a test system. Part 3 of this redbook provides a working example to guide you through the implementation of an e-commerce site.

1.4 Patterns for e-business Web site

The Patterns for e-business are published on IBM developerWorks, and can be located at: <http://www.ibm.com/software/developer/web/patterns/>. This Web site is designed to step you through the logical process of using the patterns to design e-business applications. We recommend that you follow the steps below:

1. Select a business pattern

2. Select an application topology
3. Review runtime topologies
Select a topology that satisfies the customer's requirements.
4. Review product mappings
Find out which products have been successfully used for the runtime topology selected in step 3.
5. Review guidelines
Review the guidelines and related links for the application topology and product mapping you selected in steps 2 and 4. The guidelines tell you how to design, develop and manage an e-business application.

At the time of writing this redbook, the patterns Web site provided content for the User-to-Business and User-to-Online Buying patterns. Materials for the other Patterns for e-business are in the process of being developed.

1.5 Application Framework for e-business

e-business interoperability has been a major catalyst for the rapid adoption of standards within the industry. IBM has developed an industry standard set of technologies, proven methodologies, and leadership products called the Application Framework for e-business.

The Application Framework for e-business is a combination of:

- Industry standards and technologies

The framework is based on multi-platform, multi-vendor standards such as, Java, CORBA and XML. It includes the client, application server, network, data and infrastructure standards that make it possible for a client to access services and data anywhere in the network. This model simplifies application development and deployment, enabling developers to write an application on their platform of choice and then deploy it somewhere else without completely rewriting the application.

- Proven methodology

You are building and buying mission-critical applications. To cost-effectively leverage your existing investments and transform your business, you need to know what works. So, we're harvesting our experience and turning it into methodology that our IBM services teams and partners can use to help you develop and deploy applications that meet the demands of your e-business environment. This on-going process

is focused in four areas: methodology for business analysis, application design, application development, and application deployment.

- Leadership products

Although there are others in the industry whose products adhere to the standards and philosophies of the framework, IBM is fielding a battle-tested portfolio of software, which is a solid foundation for e-business applications.

The Patterns for e-business leverage the Application Framework for e-business by making it easy to apply the technologies, standards, and products for e-business solution.

Key Architecture Elements

Figure 2 provides a summary of the architecture elements of the Application Framework for e-business.

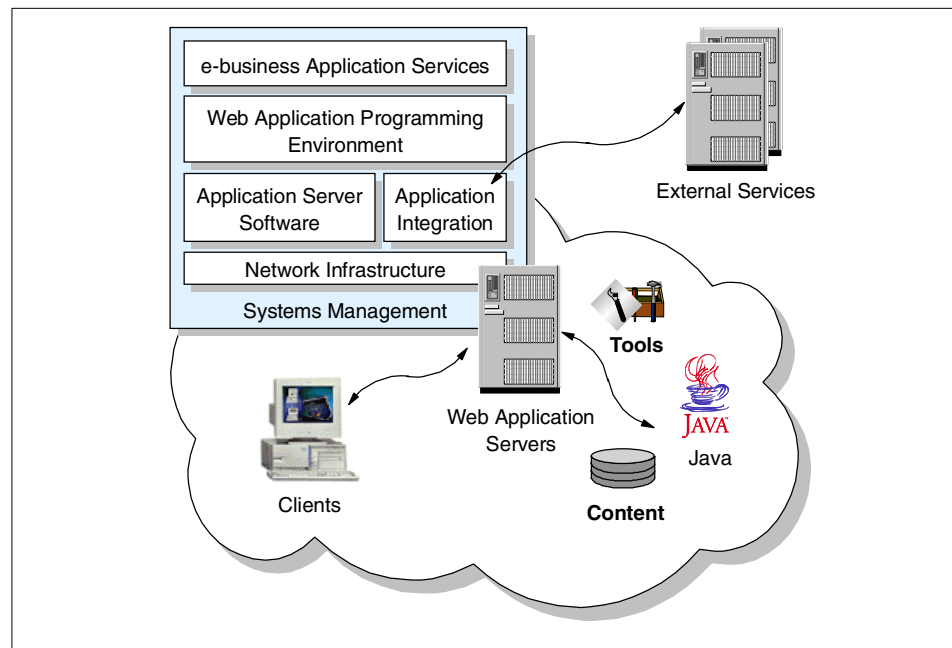


Figure 2. Architecture - Application Framework for e-business

- Clients

Clients based on the Web browser model enable universal access to Framework applications and on-demand delivery of application

components. The supported standards include HTML, Dynamic HTML, XML, and Java applets.

- Network Infrastructure

Network infrastructure provides a platform for the entire e-business environment and includes TCP/IP and network services, security services, directory services, and file and print services. The supported standards include TCP/IP, CDSA, SSL, IPsec, x.509v3 certificates, LDAP, AFS/DFS, and IPP.

- Application Server Software

Application server software provides the core function for developing and supporting the e-business application logic. This includes HTTP servers, mail and community services, groupware services, database services, transaction services, and messaging services. The supported standards are SMTP, POP3, IMAP4, IRC, NNTP, FTP, iCalendar, ODBC, DRDA, and CORBA OTS/IIOP.

- Application Integration

The Web enables e-businesses to leverage existing I/T investments. Application integration also allows disparate applications, potentially written in different programming languages and built on different architectures, to communicate with each other within an enterprise and across enterprises.

- Web Application Programming Environment

The Web application programming environment is based on Java Server Pages, Java Servlets, Enterprise Java services, and Enterprise JavaBean components, and provides an environment for writing dynamic, transactional, secure business applications on Web application servers.

- e-business Application Services

e-business application services are higher-level application-oriented components, such as e-commerce services, that conform to the Application Framework for e-business programming model. These services allow e-business solutions to be developed faster with higher quality.

- Systems Management Services

Support the end-to-end management across networks, systems, middleware, and applications.

- Development Tools

Development tools enable the creation, deployment, and management of e-business applications. It also supports integrating third party tools into the development process.

The Application Framework for e-business white papers are an important source of information included in the Patterns for e-business. The Application Framework for e-business Web site at: <http://www.ibm.com/software/ebusiness> provides a library section with a series of white papers.

1.6 An integrated view of e-business solutions

The potential usage of several Patterns for e-business may be used together for more complex system architectures.

Figure 3 shows e-business as an integration of many application domains into systems that connect a business with its customers, partners, and suppliers.

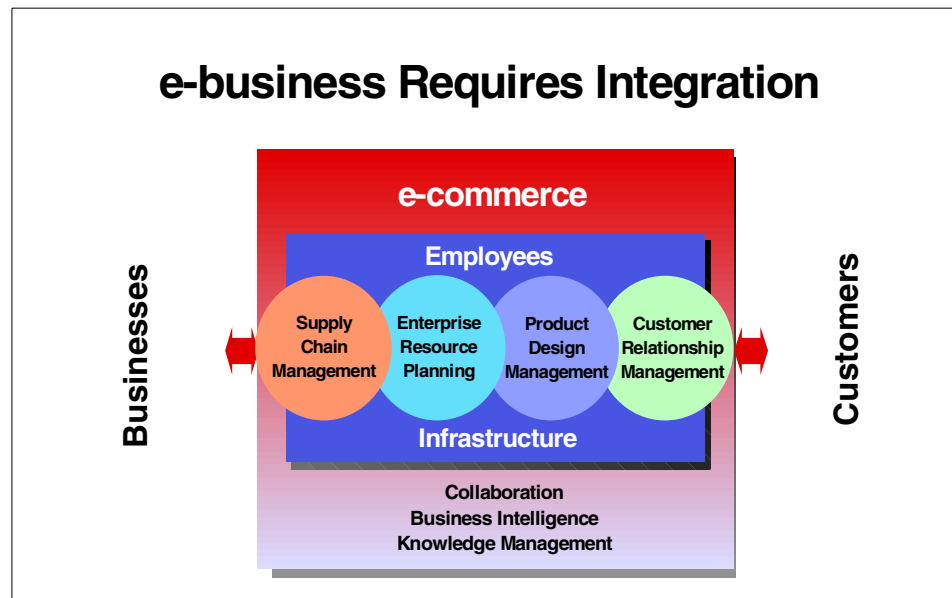


Figure 3. e-business integration

These systems are not confined to Web interfaces, although increasingly many of the user interfaces to the combined system will use Web technology.

The common set of node descriptions in the Patterns for e-business enable communication between architects and designers from very different

application domains and will suggest areas for shared nodes and infrastructure.

This is similar to the process of using design patterns to solve a programming design problem, where classes in the composed pattern play multiple roles, derived from the source patterns (see *Pattern Hatching - Design Patterns Applied* by J. Vlissides). It is different, however, in that design pattern composition is based on class diagrams and white box by nature, whereas composing architectural patterns is more component-based.

The Patterns for e-business may be applied to e-business solution areas. Table 1 provides a guide to where you may find the patterns most applicable.

Table 1. Patterns for e-business and e-business solution

e-business solution area	Business pattern
Customer relationship management	User-to-Business Pattern
e-commerce	User-to-Online Buying Pattern
Supply chain management	Business-to-Business Pattern
Collaboration	User-to-User Pattern
Business intelligence, knowledge management	User-to-User Pattern
Business application integration	Application Integration Pattern

This redbook will focus on the User-to-Online Buying Pattern.

1.7 Structure of this User-to-Online Buying Pattern redbook

Part 1 User-to-Online Buying Pattern

Chapter 2, “Choosing the application topology” on page 13, introduces application topologies 1 and 2 for the User-to-Online Buying Pattern. These application topologies capture the essential “shape” of the application solution.

Chapter 3, “Choosing the runtime topology” on page 21, discusses the runtime topologies for these application topologies. It includes discussion of variations of these topologies that are appropriate for scalability and availability.

Chapter 4, “Product mapping” on page 39, provides sample product mappings to populate the logical runtime topologies. The focus remains on the operational aspects of the solution.

Part 2 User-to-Online Buying Pattern: Guidelines

Chapter 5, “Performance guidelines” on page 53, introduces performance guidelines by considering the components of a User-to-Online Buying solution that are particularly relevant to performance.

Chapter 6, “Technology options” on page 77, discusses the technology options available to implement a Web application and provides advice on the appropriate usage.

Chapter 7, “Application design guidelines” on page 105, considers the functional components of the application within the context of the runtime topologies.

Chapter 8, “Application development guidelines” on page 153, provides guidelines for application development, considering the roles, processes and tools that are required.

Chapter 9, “System management and security guidelines” on page 187 looks at the asset management, security, and availability aspects of an e-business application.

Part 3 A working example

Part 3 of this redbook provides a working example for implementing an application topology 2 (Enterprise-out) e-commerce site.

Chapter 2. Choosing the application topology

Once the business pattern has been selected, it is time to choose an application topology. The application topologies use logical nodes to illustrate various ways to configure the interaction between users, applications and data. The chosen application topology is later mapped to a runtime topology by mapping the logical nodes to the runtime nodes.

An application topology shows the principal layout of the application, focusing on the shape of the application, the application logic and the associated data. It does not show middleware, files, or the databases where Web pages may be stored.

There are two basic approaches for application topologies:

- Web-up

The premise is to very quickly enable Web-based buying without close integration with back-end systems.

- Enterprise-out

The idea is to extend an existing order processing system to a new Web-based buying channel. In this case, there will be very close integration and re-use of back-end systems.

This redbook will provide guidelines for the Web-up and Enterprise-out topologies for the User-to-Online Buying Pattern, illustrated in application topology 1 and 2 respectively.

- Application topology 1 (Web-up)
- Application topology 2 (Enterprise-out)

The application topologies will be implemented by using the following information:

- Business drivers
- Key features
- Considerations

Choose the application topology that best fits your requirements with regard to users, applications, and data.

2.1 Application topology 1

User-to-Online Buying application topology 1 is applicable in a situation where you are building a new application and there is no need to interface with legacy or third-party applications or data. All of the required data is handled by the new e-business application. It does not need to share, access or exchange data with other applications. The application implements all required functions and does not rely on functionality provided by other existing applications. Application topology 1 in Figure 4 illustrates a three-tier Web-up commerce application with deferred access to the third tier.

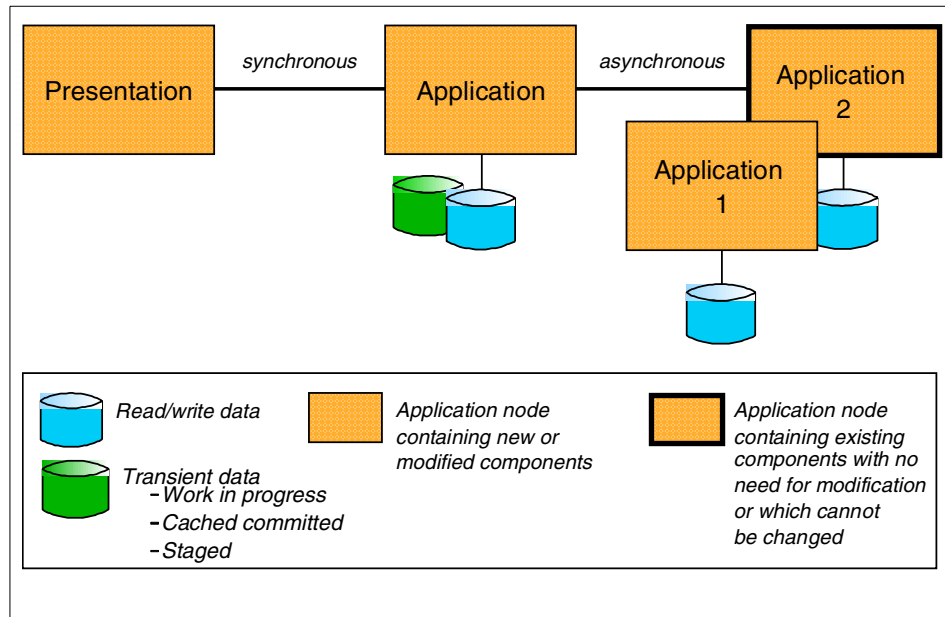


Figure 4. Application topology 1 (Web-up)

2.1.1 Application topology 1: business driver

This topology is used by corporations that may have extensive existing systems, but do not wish to invest the time to make the modifications or interfaces necessary to allow those systems to be used directly by the new Web-based online buying channel. This topology provides some other method to submit and confirm orders, either manual or batch.

The primary reasons for this approach are speed to market and cost of implementation. The objective is to get the front-end shopping and buying site

implemented quickly and postpone worrying about smooth interfaces to the back-end, which is a classic Web-up approach.

2.1.2 Application topology 1: key features

Application topology 1 represents a target topology for many Web-up application server vendors. It provides the ability for one or more point to point connections to back-end legacy applications or databases. In the future the e-commerce application can be more easily integrated with existing back-end applications. For example an ERP, or inventory management system. The examples reviewed for this application topology typically involve some form of deferred interaction with the back-end processing of the orders.

2.1.3 Application topology 1: considerations

As your e-commerce (online-buying) site becomes more successful you will most likely need better integrated interfaces to your back-end systems. You may also demand more real-time functionality, such as immediate delivery confirmation. The site will probably evolve into a Topology 2 type of implementation as described in 2.2, "Application topology 2" on page 15.

When developing an application topology 1 e-commerce solution many vendors promote ease of development by mixing the presentation and business logic layers. This type of development should be avoided, as separation promotes the effective use of discrete skill sets and code reuse. Failing to separate the business and presentation logic can lead to code that is hard to maintain and extend.

2.1.4 Application topology 1: example

A major retail department store wishes to extend their reach to a Web sales channel. As competition is keen, time to market is crucial. Existing infrastructure includes a telephone sales channel and a back-end order processing system that runs in batch. It uses *green screen* point-of-sale terminals for online transactions. In the short-term, modifications to the back-end are not feasible. The department store chooses application topology 1 because this topology does not require back-end modification.

2.2 Application topology 2

Application topology 2 for the User-to-Online Buying Pattern applies to scenarios where there is the need for integration with legacy or third-party applications. The new application is not built as a stand-alone solution, but instead needs to integrate with existing applications. The integration can be

achieved on a functional basis or a data basis using a transaction interface or a data interface.

When trying to integrate with an existing application, it is important to research whether the application needs to be modified, and determine if the changes are allowed. If the new application is going to use existing data you must determine if the data can be used “as is” or if the structure needs to be changed, and if so, is this modification possible.

Note
In most cases, the objective is to leave the existing applications and data unchanged.

Application topology 2 in Figure 5 on page 16 illustrates an Enterprise-out three-tier commerce application with online access to the third tier.

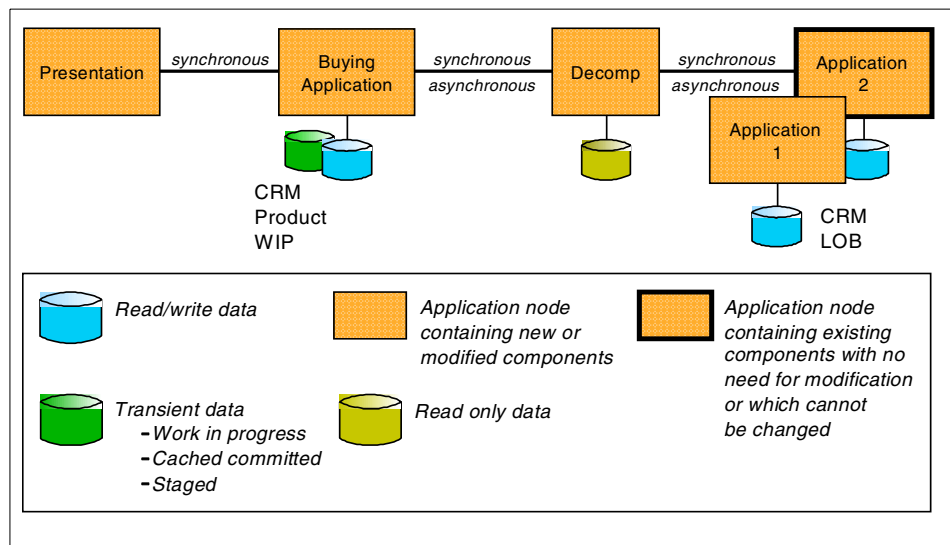


Figure 5. Application topology 2 (Enterprise-out)

2.2.1 Application topology 2: business driver

This topology utilizes Web technology to sell goods directly to the consumer. It employs the *Enterprise-out* philosophy. Wherever possible, existing order management and fulfillment systems are re-used. As organizations become more sophisticated in their implementation, they typically enable personalization. For example, they may customize the user interface, utilizing

the shopper's buying patterns or stated preferences. Sometimes this customization is referred to as *mass customization to a market of one* or personalization. Companies may *push* information to the user by prompting the shopper with information about additional purchases, specifically targeted at that individual's tastes, often referred to as *up-sell* or *cross-sell*.

2.2.2 Application topology 2: key features

Application topology 2 represents the target topology for those organizations that want to integrate the online buying application with their existing enterprise systems. The topology also addresses maintenance of data on that tier, such as:

- Product and catalog data, supporting the shopping process.
- Customer-related data for personalization and registration. The data is often duplicated in the enterprise systems on the fourth tier.
- Work-in-process data, such as data supporting shopping cart functionality, prior to submitting the orders for processing.

In addition, the pattern provides the ability to use some form of middleware to access enterprise systems on the fourth tier. The middleware routes the messages to back-end systems. The middleware may decompose a request into several back-end interactions, and may serve as a broker, potentially communicating with the systems of other companies.

Finally, the pattern supports access to back-end fulfillment systems that perform functions such as inventory, order management, pricing, shipping, tax calculation, and credit processing.

At this point, the company has built an online buying site with integration to back-end systems. Some of the legacy applications do not change, as denoted by the broad black border in Figure 5 on page 16. Some of the newly written applications are intended to support not only the Web channel, but also other sales channels such as call centers.

The interactions between the shopper and the middle-tier application will be synchronous during the buying process. Often, asynchronous confirmation messages are sent back to the user, usually by e-mail.

Between the middle-tier and the back-end fulfillment systems, the recommended approach is online synchronous interaction. This allows immediate confirmation of important information such as the availability of inventory and delivery dates. Asynchronous (but near-time) interaction is also utilized in many cases. Automated and integrated batch interaction is a less

preferable, which is sometimes the only option available. While not providing immediate confirmation for the shopper, the Enterprise-out application topology 2 is more automated than the Web-up approach.

2.2.3 Application topology 2: considerations

The same considerations mentioned in 2.1.3, “Application topology 1: considerations” on page 15, apply to application topology 2. Additionally the following concerns need to be taken into account:

- If the developer needs to provide access to many applications on the third tier, an advanced application topology may be more appropriate. You should consider an application topology that exploits a hub-and-spoke architecture between the second and third tiers.
- You should consider how this topology will be deployed to avoid systems management complexity. Complexity arises when frequently updated corporate data resides on more than one tier, with the second and third tiers both within the same organization but physically distributed. For example, synchronization of backups may be cumbersome.
- If there are different I/T organizations developing the new application and maintaining or changing the existing systems, the development might be difficult to coordinate. This is especially difficult if the interfaces between the new and the existing systems are not properly defined and documented.
- The new application might require changes to existing production systems. This is always a critical task, especially when the back-end systems or third-party applications are mission critical. Building a test system for the existing applications that needs to be changed is a good way to avoid production system failure, while developing and testing the new e-business application.
- Obtaining skilled resources that can change the third tier application is often a problem. Many times, these are old applications and finding someone who understands them well enough to change them may be difficult.
- The creation and management of the data on the second tier, and any required synchronization with the back-end systems, is often a major effort.
- The security of any customer data residing on the second tier is also a critical factor.

2.2.4 Application topology 2: example

A clothing retailer has a large hard-copy catalog business, complete with a telephone sales channel. Customers can call telemarketers to place orders with immediate confirmation of inventory, delivery dates, and prices. The back-end order processing systems are available *24-by-7*. The retailer wants to expand this business to the Web, but does not want a Web customer to have a lower level of service than a phone customer. Online integration with the order processing system will be crucial for the Web channel. Application topology 2 is an excellent choice for this retailer. It allows full integration with the back-end systems and a full set of customer services.

Chapter 3. Choosing the runtime topology

Application topologies 1 and 2 represent a starting point for delivering e-business solutions. In application topology 1 (Web-up), there is no interaction with legacy back-end systems and Web pages are served by a one or more Web servers. In the more complicated scenario, application topology 2 (Enterprise-out), there are connections to legacy back-end systems.

Once the application topology has been chosen, it is time to select the runtime topology. The runtime topology uses nodes to group functional and operational components. The nodes are interconnected to solve a business problem. An application topology leads to an underlying runtime topology.

This chapter will discuss two runtime topologies corresponding to application topologies 1 and 2. These runtime topologies are based on the Enterprise Solution Structure (ESS) Thin Client Transactional Pattern and are a representative solution for the User-to-Online Buying Pattern. For more information on ESS see “Enterprise Solutions Structure”, in the *IBM Systems Journal*, Volume 38, No. 1, 1999, found at: <http://www.research.ibm.com/journal/sj38-1.html/>.

Each topology has several variations:

- Proven basic topology
- Proven variation 1
- Emerging variation 2
- Emerging multi-tier variation 3

A variation that is labeled *proven* means that it is based on technology that has been around a while and has been the chosen method in many production systems.

An *emerging* variation is one that is based on newer technology that may or may not have been proven in production environments, but has significant benefits and is worth considering.

You may find that, depending on the customer requirements, you will need to extend the variations or combine them to get the desired results.

Most references to vertical scalability in this chapter assume upgrading the power, number or memory capacity of the processors for a given platform. Please bear in mind that one of the benefits of the Application Framework for

e-business is that vertical scalability to a big iron solution is also possible by migrating the application to another Application Framework for e-business supported platform, such as OS/390, OS/400, or RS/6000 SP.

3.1 Node types overview

The runtime topologies will be shown in graphical form in the following sections. Each topology will consist of several nodes, describing the function represented on that node. Most topologies will consist of a core set of common nodes, with the addition of one or more nodes unique to that topology. To understand the runtime topologies, you will need to review the following node definitions.

3.1.1 Web application server node

A Web application server node is an application server that includes an HTTP server (also known as a Web server) and is typically designed for access by HTTP clients and to host both presentation and business logic.

The Web application server node is a functional extension of the informational (publishing-based) Web server. It provides the technology platform and contains the components to support access to both public and user-specific information by users employing Web browser technology. For the latter, the node provides robust services to allow users to communicate with shared applications and databases. In this way it acts as an interface to business functions, such as banking, lending, and HR systems.

This node would be provided by the company, on company premises, or hosted inside the enterprise network and inside a Demilitarized Zone (DMZ) for security reasons. In most cases, access to this server would be in secure mode, using services such as SSL or IPSEC.

In the simplest design, this node can provide the management of hypermedia documents and diverse application functions. For more complex applications or those demanding stronger security, it is recommended that the application be deployed on a separate Web application server node inside the internal network.

Data that may be contained on the node includes:

- HTML text pages, images, multimedia content to be downloaded to the client browser
- JavaServer Pages

- Application program libraries, for example, Java applets for dynamic downloading to client workstations

3.1.2 Public Key Infrastructure (PKI) node

A Public Key Infrastructure (PKI) node is a collection of standards-based technologies and commercial services to support the secure interaction of two unrelated entities (for example, a public user and a corporation) over the Internet. In the context of the topologies defined in this redbook, PKI supports the authentication of the server to the browser client, using the SSL protocol.

3.1.3 Domain Name System (DNS) node

The DNS node assists in determining the physical network address associated with the symbolic address (URL) of the requested information. The DNS is that of the Internet service provider, although DNS is implemented on the accessed site, too.

3.1.4 User node

The user node is most frequently a personal computing device (PC, etc.) supporting a commercial browser, for example, Netscape Navigator or Internet Explorer. The level of the browser is expected to support SSL and some level of DHTML. Increasingly, designers should also consider that this node may be a pervasive computing device, such as a Personal Digital Assistant (PDA).

3.1.5 Directory and security services node

The directory and security services node supplies information on the location, capabilities and various attributes (including user ID/password pairs and certificates) of resources and users, known to this Web application system. The node may supply information for various security services (authentication and authorization) and may also perform the actual security processing, for example, to verify certificates. The authentication in most current designs validates the access to the Web application server part of the Web server, but it can also authenticate for access to the database server.

3.1.6 Database server node

The database server node's function is to provide a persistent data storage and retrieval service in support of the user-to-business transactional interaction. The data stored is relevant to the specific business interaction (for example bank balance, insurance information, and current purchase by the user).

It is important to note that the mode of database access is perhaps the most important factor determining the performance of this Web application, in all but the simplest cases. The recommended approach is to collapse the database accesses into a single call or very few calls. This can be achieved via coding and invoking stored procedure calls on the database.

3.1.7 Protocol firewall and domain firewall nodes

Firewalls provide services that can be used to control access from a less trusted network to a more trusted network. Traditional implementations of firewall services include:

- Screening routers (the protocol firewall in this design)
- Application gateways (the domain firewall)

The two firewall nodes provide increasing levels of protection at the expense of increasing computing resource requirements. The protocol firewall is typically implemented as an IP router, while the domain firewall is a dedicated server node.

3.1.8 Dispatcher node

The load balancer or dispatcher node provides horizontal scalability by dispatching HTTP requests among several, identically configured Web servers.

3.1.9 Shared file system node

The timely synchronization of several Web servers is achieved by using a shared file system as the content storage and capitalizing on the replication capability of this technology.

3.1.10 Web server redirector node

In order to separate the Web server from the application server, a so-called *Web server redirector node* (or just redirector for short) is introduced. The Web server redirector is used in conjunction with a Web server. The Web server serves HTTP pages and the redirector forwards servlet and JSP requests to the application servers. The advantage of using a redirector is that you can move the application server behind the domain firewall into the secure network, where it is more protected than within the DMZ. Static pages can be served from the DMZ by this node.

The redirector can be implemented, for example, by either a reverse proxy server or by a Web server plug-in such as the servlet redirector function of IBM WebSphere Application Server Advanced Edition.

3.1.11 Application server node

This node provides the infrastructure for application logic and may be part of a Web application server node. It is capable of running both presentation and business logic but generally does not serve HTTP requests. When used with a Web server redirector the application server node will run both presentation and business logic. In other situations, it may be used for business logic only.

3.1.12 Commerce server node

This node provides the infrastructure for application logic and may be part of a Commerce server node. It is capable of running both presentation and business logic.

3.1.13 Application and data nodes

Existing applications are run and maintained on nodes that are installed in the internal network. These applications provide for business logic that uses data maintained in the internal network. The number and topology of these existing application and data nodes is dependent on the particular configuration used by these legacy systems.

Where do you go from here?

In Chapter 2, “Choosing the application topology” on page 13, you were assisted in choosing the appropriate application topology for your environment.

If you chose application topology 1, continue to 3.2, “Runtime topology A” on page 25.

If you chose application topology 2, then skip the next section and go directly to 3.3, “Runtime topology B” on page 32.

3.2 Runtime topology A

Runtime topology A is based on the IBM Enterprise Solution Structure (ESS) Reference Architecture and relates to application topology 1 (Web-up).

In Figure 6 on page 27, is composed of logical nodes that represent a collection or grouping of the major building blocks of the solution. Nodes are used by the lead architect to describe the design at a logical level. They contain a set of components that provide either infrastructure or application functionality. Examples of infrastructure components that could run on a commerce server node include Web servers, Java Virtual Machines, and TCP/IP stacks. Examples of application components that could reside on a commerce server node include HTML pages, Java Servlets, and CGI programs developed by the programming team. Logical nodes map to product instantiations in a step called *product mappings*. In some cases the logical node will map one-to-one to a physical machine; however, this is not always the case. Multiple nodes may be mapped to one machine, or a single logical node may be spread across multiple physical machines.

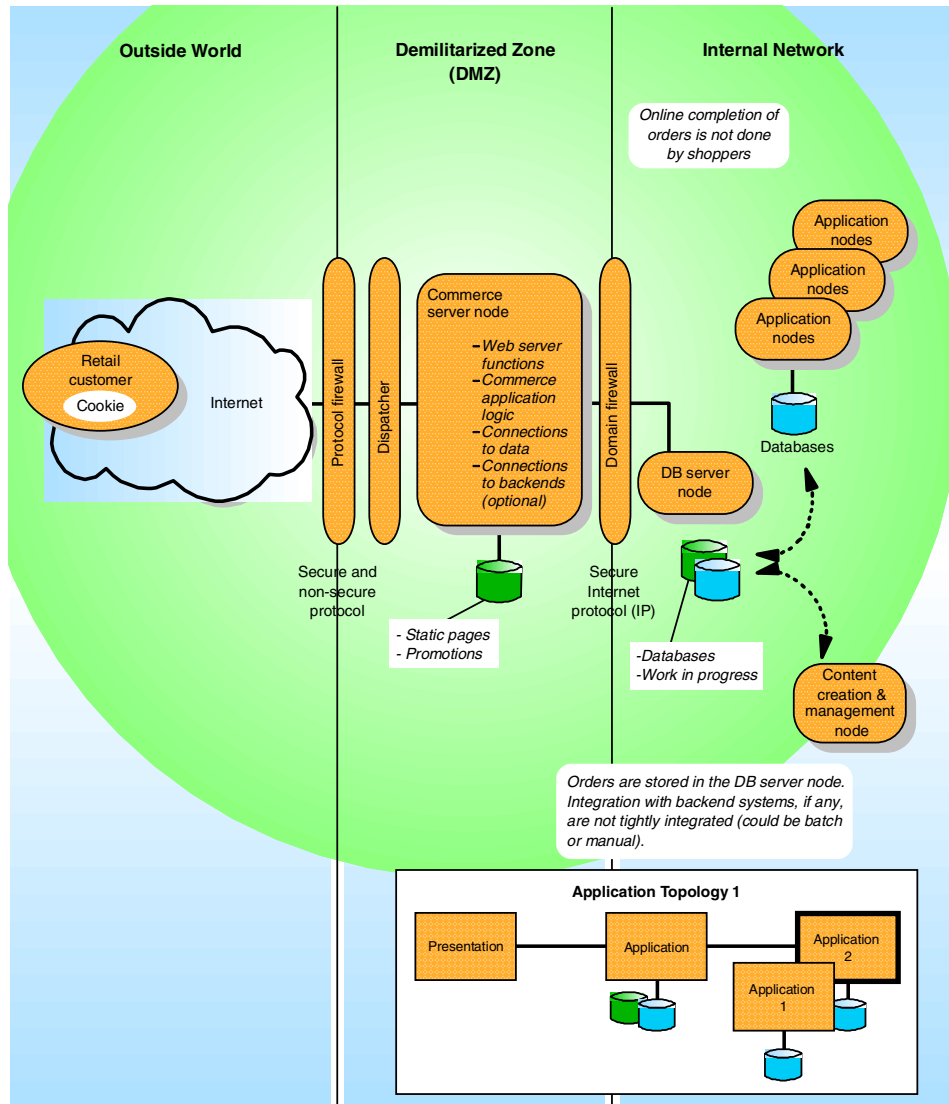


Figure 6. Runtime topology A - application topology 1 (Web-up)

At the logical level, this runtime topology involves a synchronous interaction with the user during the first part of the shopping experience. An asynchronous (and deferred) response for the processing of details and confirmation of a submitted order follows. Therefore, this runtime does not allow for such real-time features as immediate inventory validation, order completion, or shipping details.

Web-up basic online shopping process

1. From a Web browser client, the shopper connects to the commerce site by entering the Web site URL, will do one of the following:
 - Log in to the commerce site if the user is recognized as a registered shopper from a profile on the database server.
 - The user may browse the site anonymously.
 - Enter profile information to become a registered user.
2. The user then interacts with the pages of the site. The pages are either static pages, served from the commerce server or dynamically built pages with information from the database server.
3. Items will be added to a shopping cart. The data for the shopping cart is stored on the database server along with required session state information. A cookie is sent to the client browser. This cookie allows the commerce server to track the progress of the customer interactions on the commerce site and connect users with their shopping cart.
4. When the shopper wishes to buy or checkout items in the shopping cart, the commerce server stores the submitted order on the database server for later processing. An acknowledgment is sent back to the shopper that the order has been submitted, and the interactive session is terminated.
5. The system will use one of several techniques to retrieve the order and process it through the back-end fulfillment, inventory, payment and shipping processes.

Note

This pattern does not specify the details of back-end integration. It may be done in batch or manually. The example given 3.2.1, "Runtime topology A: Example" on page 29, describes how it performed order processing functions.

6. Eventually, a confirmation e-mail of order status (such as delivery, out of stock, credit problems) is sent to the user.

Typically, the logon process uses registration data from the database server node to identify registered shoppers and encryption technology, such as SSL, to protect sensitive data (for example, credit card or address information).

Currently, most sites do not use digital certificates. In application topology 2, the client is focused on extending the back-end systems with a tightly integrated Web access channel; integration with corporate-wide directory and

security mechanisms may be desired. The runtime topology B corresponding to application topology 2, includes directory and security nodes on the internal network as seen in Figure 8 on page 33.

In the runtime topology A, the Directory and Security nodes are omitted, since corporate-wide directory and security mechanisms are usually not desired. The “ESS Reference Architecture”, used by IBM Global Services, provides a much more detailed flow of the online shopping process and how it interacts with the infrastructure. This information is often used to prepare vendor quotes for the products necessary to physically implement the logical runtime topology.

3.2.1 Runtime topology A: Example

In 2.1.4, “Application topology 1: example” on page 15, the example retail department store needed to expand its sales to the Web. The example retail store chose application topology 1 of the User-to-Online Buying Pattern because of the need for a Web-up solution.

The retail department store implemented support for two distinct buying channels within the same site:

- Web browser-based shoppers use the Web channel
- Telemarketers utilize their point-of-sales system to complete orders

Walking through a purchase

1. During the shopping process, the user interacts synchronously with the commerce server and the database server. This process is similar to the shopping experience found in the *Enterprise-out* approach to the topology 2 runtime. Once the shopper has decided to buy the items in the shopping cart, this pattern takes a very different approach:
 - The shopper is simply notified that the order has been taken and that confirmation is forthcoming
 - Most importantly, there is no online linkage from the online buying front-end to the back-end systems (manual).
2. The lower part of Figure 7 on page 31 shows how order processing is completed:
 - In the past, the orders were processed by telemarketers by phone. With the new Internet front-end, the call center representatives simply log in to the new application on commerce server and access any submitted, but unfulfilled, orders on the database node.

- On behalf of the online shopper, the call center representatives submit the unfulfilled orders to the back-end order processing systems. These systems send e-mail to the shoppers, confirming their orders and providing such information as delivery dates. While the online shopper is interacting with the system, they are unaware of factors such as, delivery dates or out-of-stock items. This is a major drawback for this topology.

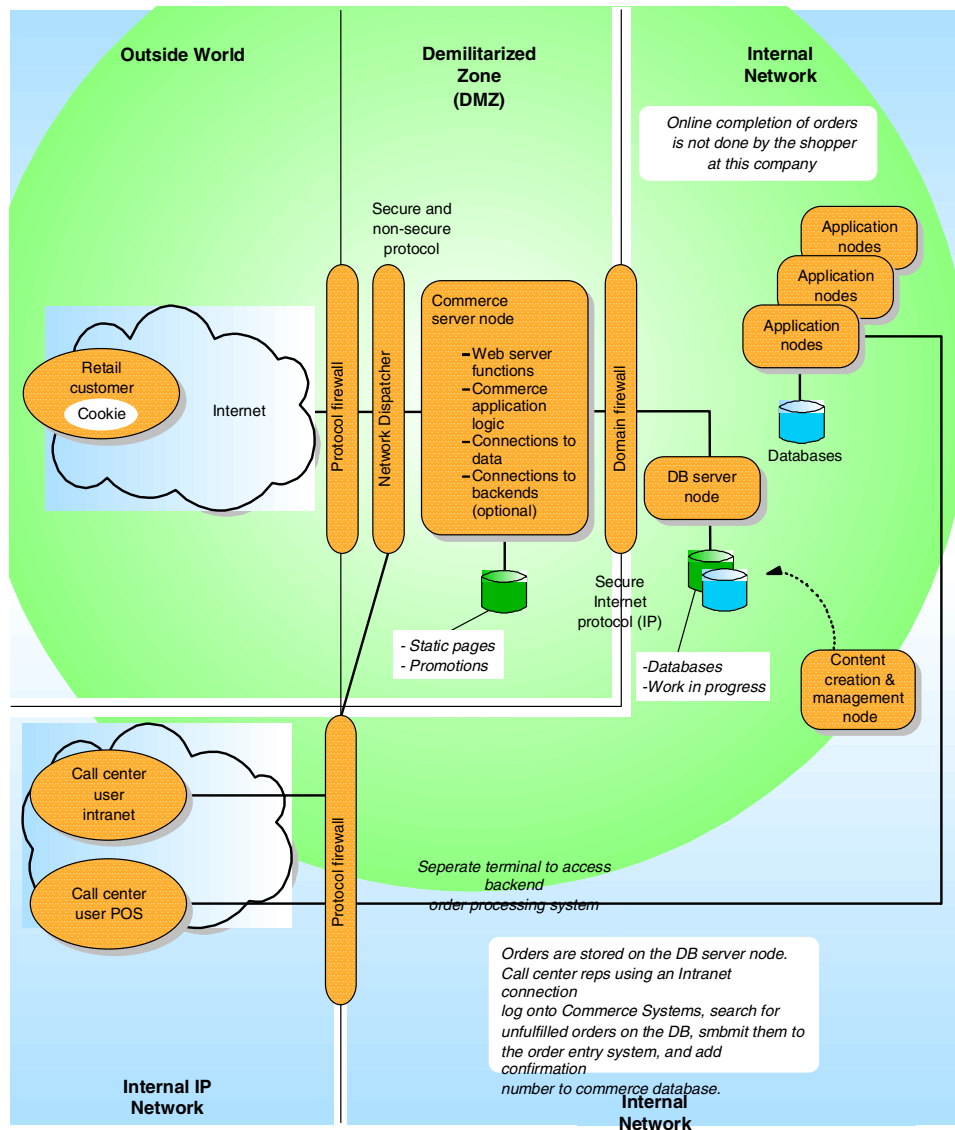


Figure 7. Runtime topology A - example for application topology 1 (Web-up)

Reasons for this approach

- Fast time to market, 2-3 months, was critical for this customer
- A process for customers to order over the phone was already in place.
- The back-end order processing systems offered no interactive interface for the commerce server front-end to utilize for direct online access.

- The back-end order processing systems did not provide the 24x7 hours of operation demanded by Internet shoppers.
- The volume of orders were not expected to be large initially, so they believed they could handle the load manually. This approach provided a quicker time-to-market than that required by integrating directly with the back-end systems. The retailer expected lower volume due to the nature of their sales: primarily big ticket items such as refrigerators, not high-volume items such as pairs of socks.

Obviously, the retailer intends to expand usage and include many more items. With augmented advertising of the site, the retailer expects sales volume to increase dramatically and faces a pressing need to provide a less manual approach to the order processing back-end systems. This site will evolve over time into one that more closely resembles an Enterprise-out solution, runtime topology B: application topology 1. It will include automated interfaces to order processing either in batch or directly online.

Where do you go from here?

If you chose application topology 1, skip the next section, go to Chapter 4, “Product mapping” on page 39.

If you chose application topology 2, continue to “Runtime topology B” on page 32.

3.3 Runtime topology B

Runtime topology B is the preferred User-to-Online Buying runtime topology. It allows for immediate confirmation of order completion, combined with information such as shipping details and volume discount pricing. This runtime topology involves either a synchronous or asynchronous connection to the back-end, with synchronous being the preferred choice. This runtime provides online access to the back-end systems during checkout and order completion. It is based on the “ESS Reference Architecture”. This runtime topology represents a good starting point for selling merchandise over the Web. Typically, it involves integration with existing back-end systems as part of the Enterprise-Out approach of Web enabling existing systems to sell goods.

Figure 8 on page 33 is composed of logical nodes that represent a collection or grouping of the major building blocks of the solution.

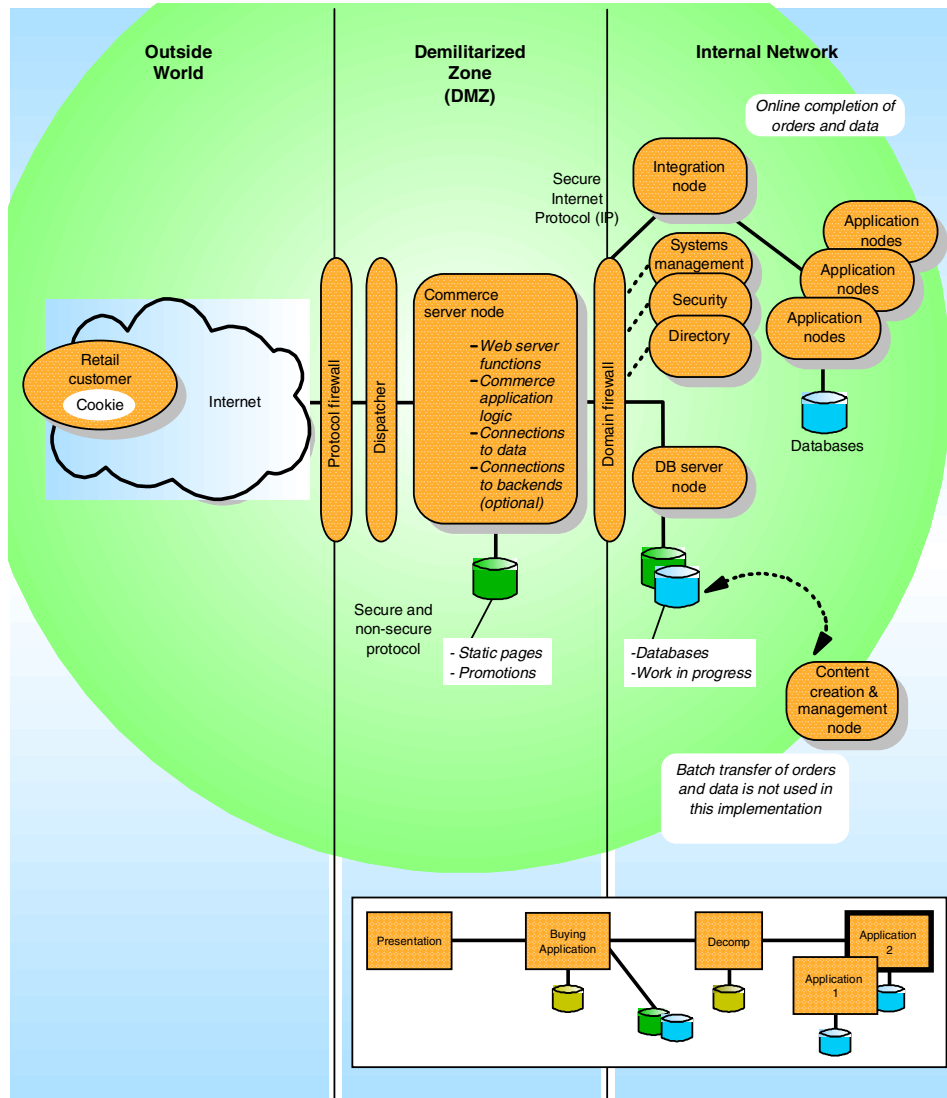


Figure 8. Runtime topology B: application topology 2 (Enterprise-out)

Enterprise-out basic online shopping process

1. From a Web browser client, the shopper connects to the commerce site by entering the site URL. At this point the customer will do one of the following:
 - Log in to the commerce site if they are recognized as a registered shopper from a profile on the database server.

- The user may browse the site anonymously.
 - Enter profile information to become a registered user.
2. The user then interacts with the pages of the site. The pages are either static pages, served from the commerce server, or dynamically built pages with information from the database server.
 3. Items will be added to a shopping cart. The data for the shopping cart is stored on the database server along with required session state information. A cookie is sent to the client browser. This cookie allows the commerce server to track the progress of the customer interactions on the commerce site and connect users with their shopping cart.
 4. When the shopper wishes to buy or check out items in the shopping cart, one of two implementations are taken:
 - a. Interaction through the integration node to access the back-end application nodes (such as order processing, pricing, inventory, credit, and shipping) and immediately provide feedback to the shopper such as confirmations and delivery dates. This is the preferred approach.
 - b. The less preferred approach is to store the submitted order on the database server for later submission via batch processing. Acknowledgment that the order has been submitted is sent back to the shopper from the commerce server to the Web browser. An e-mail confirmation of such events as delivery, out of stock conditions, and credit problems will be sent later by the back-end processing systems. This approach is less desirable from a user's viewpoint because it does not provide such features as immediate inventory validation, order completion, and estimated shipping details. It is less desirable from the company's viewpoint because it allows less direct control of such things as order completion and shipping processes. This approach is similar to the Web-up approach, except that it provides more automated linkage to the batch processes.

Typically the logon process uses registration data on the Database node to identify registered shoppers and encryption technology, such as SSL to protect sensitive transmissions (for example, credit card or address information). Currently, most sites do not use digital certificates. This runtime topology does include directory and security nodes, indicating that future use of technologies, such as LDAP directories and digital certificates is likely. The content creation and management node represents the functions required to create, and administer the database server, and commerce server nodes.

The ESS Reference Architecture used by IBM Global Services provides a more detailed flow of the online shopping process and how it interacts with

the infrastructure. This information is often used to prepare vendor quotes for the products necessary to physically implement the logical runtime topology.

3.3.1 Runtime topology B: Example

Each project team may need to modify the basic node diagram of how the logical architecture works in order to accommodate individual client requirements. This example shows modifications made to produce a client-specific diagram for a customer site as seen in Figure 9 on page 36.

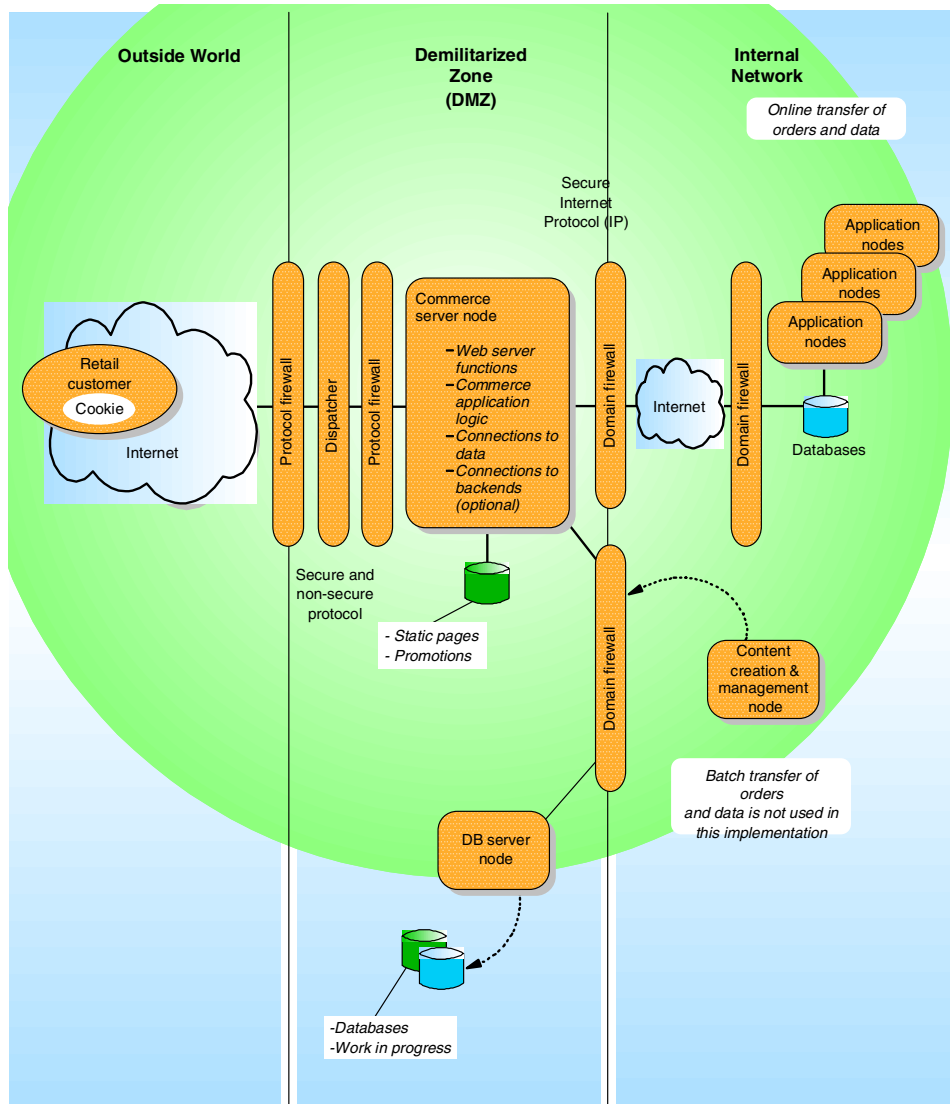


Figure 9. Runtime topology B- example for application topology 2 (Enterprise-out)

The example outsourced commerce front-end is connected via a synchronous link to in-house back-end order processing systems (Enterprise-out approach).

In 2.2.4, “Application topology 2: example” on page 19, the clothing retailer need to expand its sales to the Web. The company intends to outsource the

Web front-end to the IBM Universal Server Farm, and have the back-end reside on the company's host systems in its data center. The IBM Universal Server Farm refers to a set of outsourcing sites run by IBM. Server technology from multiple vendors is operated by IBM on behalf of organizations that would rather not manage their own server operations. Frequently in Online Buying sites, the nodes in the DMZ front end and the database server node are outsourced. These are then networked to the company's own data center for back-end order processing.

The reasons for implementing their Web channel with the Enterprise-out runtime topology B are discussed in "Reasons for architectural approach" on page 37. This approach is quite close to the basic runtime topology B. It differs in the following ways:

- The front-end commerce site is outsourced and hosted at the IBM Universal Server Farm.
- The back-end resides at the company's in-house data center. The modification is that communication between the DMZ and the internal network takes place over the Internet using a secure connection between two firewalls (firewalls 4 and 5). This also implies TCP/IP communication to the back-end.
- The retailer chose the preferred approach of linking to the back-end order entry systems in a synchronous and online manner. The reasons for this are listed in "Reasons for architectural approach" on page 37.
- A separate batch link exists for publishing and maintaining data on the database that is necessary for administration of the site.

Reasons for architectural approach

The company had a mandatory business requirement that users have immediate confirmation of orders with shipping, discounts, and pricing information. This data was to be provided through the back-end order processing systems. The company was providing immediate confirmation of call center orders and did not want less functionality for its Web customers. The company needed to support the existing call center. The back-end systems were already in place with server interfaces, and sufficient hours of operation. It was relatively easy to utilize the same interfaces to support Internet access.

Chapter 4. Product mapping

In this chapter we map the logical nodes defined in the runtime topology to products for the selected platform. The product mapping identifies the platform, software product name, and version. The IBM Application Framework for e-business provides support for many platforms, including IBM AIX, IBM OS/400, IBM OS/390, Sun Solaris, HP-UX, Linux, and Windows NT/2000.

The framework provides the flexibility to allow you to develop and test the application on your runtime platform and easily deploy the application on the customer's platform of choice.

Note

It is common for a company to have a mixture of platforms within the whole integrated e-business solution. The requirement for integration with a mixed platform environment makes the Application Framework for e-business a very appealing choice with its support for many platforms.

In order to perform the product mapping, we need to first select a platform. When selecting a platform, you should consider the following points:

- Existing systems and platform investments
- Available customer and developer skills
- Customer choice
- The platform selected should fit into the customer's environment and ensure quality of service, such as scalability and reliability to ensure the solution can grow with the e-business.

Next, we will review the product mapping. This redbook provides information on the Using WebSphere Commerce Suite product mapping for Windows NT and AIX platforms. Information on product mapping for other platforms can be found on the Patterns Web site at:

<http://www.ibm.com/software/developer/web/patterns/>

Where do you go from here?

In Chapter 3, “Choosing the runtime topology” on page 21, you were assisted in choosing the appropriate runtime topology for your application.

If you chose runtime topology A, continue to 4.1, “Runtime topology A: product mapping” on page 40.

If you chose runtime topology B, then skip the next section and go directly to 4.2, “Runtime topology B: product mapping” on page 43.

4.1 Runtime topology A: product mapping

This section provides product mapping information for runtime topology A logical nodes, for specific software products on the selected platform.

4.1.1 Runtime topology A: product mapping - Windows NT platform

Figure 10 on page 41 illustrates the NT platform software product names, and versions typically used in a Using WebSphere Commerce Suite for runtime topology A.

4.1.2 Runtime topology A: product mapping - AIX platform

Figure 11 on page 42 illustrates the AIX platform software product names, and versions typically used in a Using WebSphere Commerce Suite for runtime topology A.

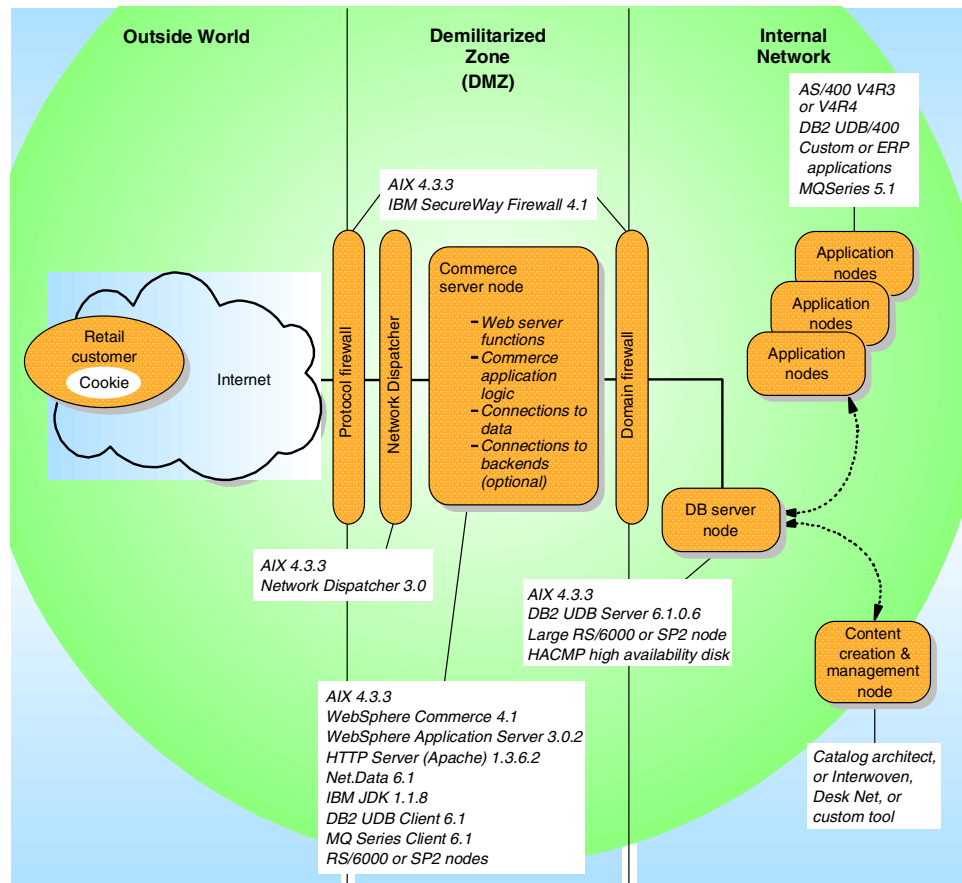


Figure 11. Runtime topology A: product mapping - AIX Platform

Product mapping example - AIX platform

In Figure 11 on page 42 illustrates the example company product mapping for the AIX platform. The company had the benefit of implementing one operating system, AIX, within the entire DMZ network. Even the firewalls were able to run on the AIX platform.

In the case of a Web-up implementation, the back-end may not exist or may be installed concurrently with the front-end systems. In such cases, the back-end application servers may also use the AIX platform.

Typically, multiple RS/6000 or SP2 commerce servers will be clustered using the Network Dispatcher for load balancing. Large enterprise customers, often are running on S/390 hardware and software for back-end processing with legacy systems.

If the commerce and database servers had been implemented using Sun Solaris or HP-UX technology, the product mapping would be similar.

Where do you go from here?

If you chose runtime topology A, skip the next section and go directly to Part 2, “User-to-Online Buying Pattern: guidelines” on page 51.

If you chose runtime topology B, continue to 4.2, “Runtime topology B: product mapping” on page 43.

4.2 Runtime topology B: product mapping

This section provides product mapping information for runtime topology B logical nodes, for specific products on the selected platform.

4.2.1 Runtime topology B: product mapping - Windows NT platform

Figure 12 on page 44 illustrates the Windows NT platform software product names, and versions typically used in a Using WebSphere Commerce Suite for runtime topology B.

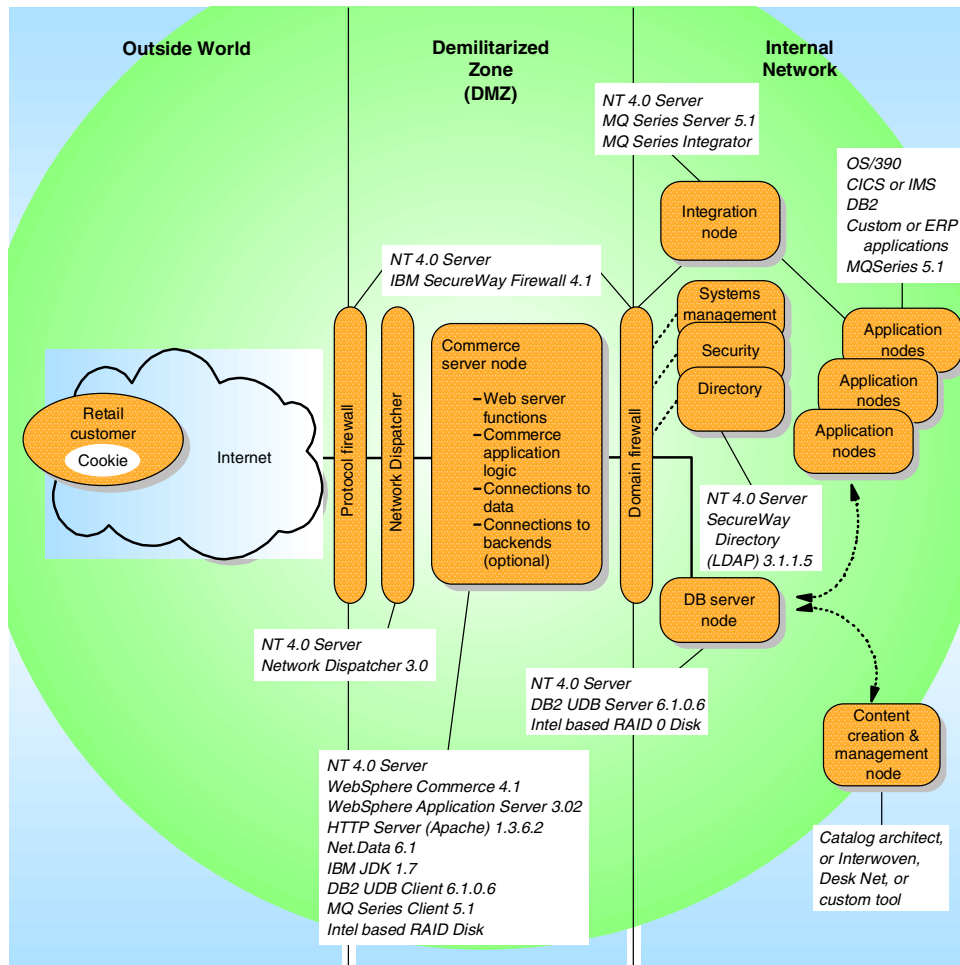


Figure 12. Runtime topology B: product mapping - Windows NT platform

4.2.1.1 Example product mapping - Windows NT Platform

In Figure 12 on page 44, the example company had a large investment in Windows NT. The company had the benefit of implementing one operating system platform, Windows NT, within the DMZ network.

4.2.1.2 Detailed product mapping - Windows NT Platform

The section provides detailed tables for the major logical nodes within runtime topology B to a specific software product name and version for the Windows NT platform.

Table 2. Commerce server node - NT platform

Product name	Version
Microsoft Windows NT Server	4.0
Microsoft Windows NT Service Pack	6a
IBM JDK	1.1.7
IBM HTTP Server (Apache)	1.3.6.2
IBM DB2 UDB Universal Database Extended Edition - client	6.1.0.6
IBM WebSphere Application Server Advanced Edition	3.02
IBM WebSphere Commerce Suite Pro Edition	4.1
IBM MQSeries - client	5.1

Table 3. Database server node - NT platform

Product name	Version
Microsoft Windows NT Server	4.0
Microsoft Windows NT Service Pack	6a
IBM DB2 UDB Universal Database Extended Edition - client	6.1.0.6

Table 4. Firewall node - NT platform

Product name	Version
Microsoft Windows NT Server	4.0
Microsoft Windows NT Service Pack	6a
IBM SecureWay Firewall	4.1

Table 5. Dispatcher node - NT platform

Product name	Version
Microsoft Windows NT Server	4.0
Microsoft Windows NT Service Pack	6a
IBM Network Dispatcher (bundled in WebSphere Edge Server 1.0)	3.0

Table 6. Directory server node - NT platform

Product name	Version
Microsoft Windows NT Server	4.0
Microsoft Windows NT Service Pack	6a
IBM SecureWay Directory (LDAP)	3.1.1.5

Table 7. Integration server node - NT platform

Product name	Version
Microsoft Windows NT Server	4.0
Microsoft Windows NT Service Pack	6a
IBM MQSeries - server	5.1

4.2.2 Runtime topology B: product mapping - AIX platform

Figure 13 on page 47 illustrates the AIX platform software product names, and versions typically used in a Using WebSphere Commerce Suite for runtime topology B.

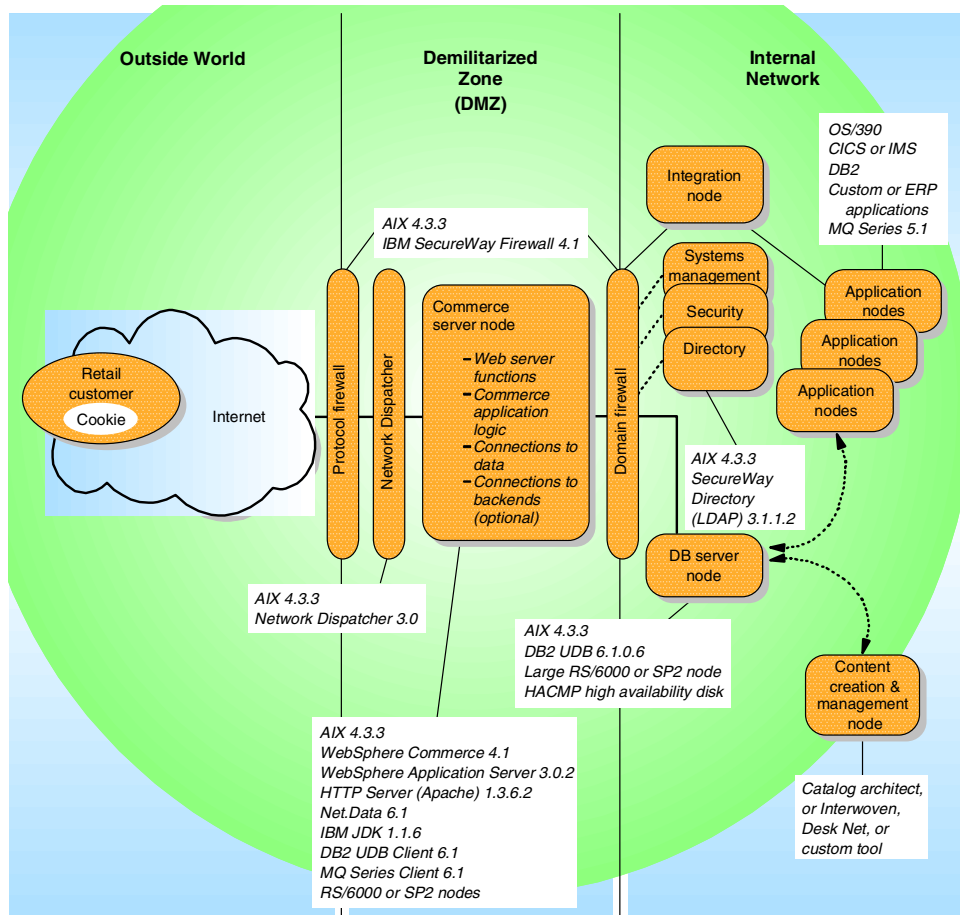


Figure 13. Runtime topology B: product mapping - AIX platform

4.2.2.1 Example product mapping - AIX platform

In Figure 13 on page 47, the example company had a large investment in the AIX platform. The company had the benefit of implementing one operating system, AIX, within the entire DMZ network. Even the firewalls were able to run on the AIX platform.

Typically, multiple RS/6000 or SP2 commerce servers will be clustered using the Network Dispatcher for load balancing. Large enterprise customers, often are running on S/390 hardware and software for back-end processing with legacy systems.

If the commerce and database servers had been implemented using Sun Solaris or HP-UX technology, the product mapping would be similar.

4.2.2.2 Detailed product mapping - AIX platform

The section provides detailed tables for the major logical nodes within runtime topology B to a specific software product name and version for the AIX platform.

Table 8. Commerce server node - AIX platform

Product name	Version
IBM AIX operating system	4.3.3
IBM JDK	1.1.8
IBM JDK 1.1.8 PTF	6
IBM HTTP Server (Apache)	1.3.6.2
IBM DB2 UDB Universal Database Extended Edition - client	6.1.0.6
IBM WebSphere Application Server Advanced Edition	3.02
IBM WebSphere Commerce Suite Pro Edition	4.1
IBM MQSeries - client	5.1

Table 9. Database server node - AIX platform

Product name	Version
IBM AIX operating system	4.3.3
IBM DB2 UDB Universal Database Extended Edition - client	6.1.0.6

Table 10. Firewall node - AIX platform

Product name	Version
IBM AIX operating system	4.3.3
IBM SecureWay Firewall	4.1

Table 11. Dispatcher node - AIX platform

Product name	Version
IBM AIX operating system	4.3.3
IBM Network Dispatcher (bundled in WebSphere Edge Server 1.0)	3.0

Table 12. Directory server node - AIX platform

Product name	Version
IBM AIX operating system	4.3.3
IBM SecureWay Directory (LDAP)	3.1.1.2

Table 13. Integration server node - AIX platform

Product name	Version
IBM AIX operating system	4.3.3
IBM MQSeries - server	5.1

Part 2. User-to-Online Buying Pattern: guidelines

Chapter 5. Performance guidelines

In Chapter 4, “Product mapping” on page 39, you mapped the specific software products and versions to the selected hardware platform. The next step is to determine the hardware capacity required for each system. When reviewing the capacity requirements, there are many performance considerations that need to be analyzed to optimize the performance of your e-business solution.

This chapter discusses the various aspects of performance and how the design and implementation of an e-commerce application can affect performance. The chapter is organized into the following sections:

- Architecture and hardware capacity planning
- Operating system and network
- Web application servers and technologies
- Application development
- General performance guidelines
- Where to find more information

5.1 Architecture and hardware capacity planning

In this section, we will highlight the performance implications of three runtime topologies for the WebSphere Commerce Suite.

5.1.1 1-tier runtime topology (single node)

In a 1-tier topology, all servers and modules of the WebSphere Commerce Suite reside on the same physical node.

This configuration is suitable for a development and test environment. As a rule of thumb, this environment should be avoided for a production environment, for performance and scalability reasons.

Note

In some cases, this may be suitable for production on a high-end AS/400 or S/390 system.

5.1.2 2-tier runtime topology (single WCS server - remote DB server)

This topology is typically used for a small commerce Web site. This configuration provides for some scalability by having a remote database server system and a separate commerce server. All application server and HTTP server processes are hosted on a single machine. This topology facilitates a more streamlined system of backups, system maintenance, and performance tuning.

The key difference between the 1-tier and 2-tier is the system architecture. The tier-2 runtime topology can easily scale by adding additional commerce servers that are load balanced by the Network Dispatcher.

5.1.3 3-tier runtime topology (multiple WCS server - remote DB server)

This topology is typically used by enterprise commerce Web sites. This is the preferred runtime topology for any company that can afford it. There will be a very big range in the cost of this architecture depending on the platforms and configuration of the hardware selected.

This configuration provides for scalability by having a remote database server system and separate commerce servers. All application server and HTTP server processes are hosted on a single machine. This topology facilitates a more streamlined system of backups, system maintenance and performance tuning.

The key difference between the tier-2 and tier-3 runtime topology is the scalability and performance gains provided by the Network Dispatcher. This configuration provides load balancing by using the IBM Network Dispatcher to redirect requests to the clustered resource of commerce servers.

5.1.4 Hardware capacity planning

The WebSphere Commerce Suite Pro, *Installation Guide* for AIX and Windows NT, packaged with the WebSphere Commerce Suite product, provide some guidelines on hardware capacity planning based on the topology you have selected. The types of systems that are listed will become obsolete as newer versions of the hardware become available. It is always best to confer with your IBM marketing representative or technical support team for the latest information on this topic.

5.2 Operating system and network

Performance tuning is very much hardware platform and operating system specific. Base operating system tuning is a very important part of optimizing the

performance of your e-business solution. Due to the vast nature of this topic, we will not cover this topic in detail in this redbook. We suggest that you refer to your hardware and operating system vendors for information on this topic.

The more traffic a commerce Web site receives, the greater the consumption of system and network resources. Increased traffic can strain the system and slow its performance, affecting the time required to complete a purchase. If a site is very large and busy, you may wish to consider one of the following solutions:

- Faster system or platform for WebSphere Commerce Suite
- Multiple Network Dispatched WebSphere Commerce Suite systems
- Separate database server system

If slow performance is being caused by too many complex SQL queries of a large database, you can place the database on a separate machine. A machine dedicated to the database alone will better handle the database queries without a performance loss, and the removal of the database from the Commerce Suite server machine will free system resources and improve performance on the server.

- Faster hardware or operating system for database server

You may also consider hosting the database on a larger computer, such as an AIX machine, and the rest of the system on a PC. If delays are caused by a large number of server requests, you can use more than one machine to serve the pages. As a general rule, if system resources are being overtaxed regularly during peak usage periods, increase physical resources to improve the performance.

On Windows NT, you can use the Performance Monitor to monitor CPU and I/O use to find resource bottlenecks. The Commerce Suite daemon is named `server.exe`, and the DB2 Universal Database engine is named `db2sysc.exe`. Consider moving the database engine to a separate machine if it is using an excessive amount of system resources. Other factors, such as tuning the database and caching pages to save system processing, will influence the size of system you require. If you use multiple machines, you will need a routing system, such as IBM's Network Dispatcher, to control workflow through the system.

- Increase system memory

If the system resources are only being slightly overtaxed, you may be able to compensate by simply increasing the amount of memory in the system. A memory increase is one of the most effective and inexpensive upgrades you can make. Other factors, such as the number of CPU's the system contains, will influence the amount of memory you require.

- Network bandwidth

Ensure that you have enough bandwidth to support the user requests on the system. A small, lightly trafficked commerce site may function well on an ISDN connection, while a very large site may require its own T3 line. You should test traffic volume regularly, and ensure that you have enough bandwidth to manage requests and serve pages without slowdown at peak periods. It is not sufficient to have only enough bandwidth to manage workload on average. Other factors, such as the size of pages, will influence the amount of bandwidth you require.

5.3 Web application servers and technologies

The information in this chapter is specific to the AIX platform, but many of the techniques apply to other operating systems. This section provides performance guidelines on the technologies comprising the following servers:

- WebSphere Commerce Suite
- Database server
- Network Dispatcher
- Back-end servers

5.3.1 WebSphere Commerce Suite

The WebSphere Commerce Suite Pro Edition is comprised of many servers and technologies that can be tuned for better performance. We cover the topics that need to be reviewed when optimizing your solution for performance. The following topics are discussed:

- Java and Java Virtual Machines
- HTTP Server
- WebSphere Application Server
- WebSphere Commerce Suite Server

5.3.1.1 Java and Java Virtual Machines

Java is an interpreted language. Java source code must first be compiled into portable bytecodes that can then be interpreted in the Java Virtual Machine (JVM) on the local system. All JVMs are not created equal, and performance should be considered when selecting a JVM.

Selecting JVMs

High-performance JVMs are critical for good performance. Here are some considerations when evaluating JVMs:

1. The JVM must be supported by the application server.
2. Java compiler and virtual machine technology change rapidly. Today a JIT version of a compiler may provide the best performance for your solution, while tomorrow it may be a static compiler, and next week there may be a new technique.
3. A JVM should be certified for portability by a recognized certification authority such as JavaSoft.
4. JVMs that have been optimized for a specific operating system tend to perform better than other JVMs.
5. Systems with symmetric multiprocessors (SMP) tend to perform better because of the thread support in Java. Use JVMs that effectively support SMP systems.
6. Evaluate the trade-off between the stability of the current release of a JVM and the performance enhancements available in the newest beta release and/or production version.
7. Some vendors provide service for the JVM only by providing a new release when it becomes available. IBM does provide defect support for IBM packaged JVMs.
8. Some JVMs implement an advanced feature called garbage collection (GC) to boost the programming productivity and to avoid the common pitfall of traditional programming languages, memory leaks. However, GCs can slow down the Java program execution because most GCs utilize a *stop and copy* technology. Due to ongoing research, GCs have been developed that do not exhibit any of the problems described above. These GCs run incrementally and take many characteristics of the Java language into account.

The performance of applications can often be better improved by improving the runtime characteristics of the algorithms used. However, there are circumstances, like computationally intensive programs, where an improvement in the performance of the underlying JVM will add to the solution's overall performance.

Just-In-Time compiler

A Just-In-Time (JIT) compiler converts bytecodes into native code on-the-fly with some optimization, and it should run much faster than just a Java Virtual Machine (JVM). The JIT is not a replacement for the JVM, but an enhancement to improve performance. A JIT works well for computationally intensive programs that execute the same segment of instructions repeatedly. But it does not yield significant improvement for programs that are I/O intensive or cause a lot of garbage collection. This is because the program

code takes up such a small amount of time and thus any optimizations would become negligible.

In addition, JITs are limited in the extent of optimizations they can do because their compile is a runtime cost. Also, since JIT compilers normally do not have the “global view” of the executed code, the code they generate is of poorer quality than the code generated by static compilers.

Adaptive compilers

Adaptive compilers try to overcome the weaknesses of JIT compilers with adaptive optimization techniques. The idea is to intelligently select, compile and optimize frequently used and/or resource-intensive portions of the code, so called “spots.” Once these spots have been determined, optimization techniques known from traditional compilers are applied to compile a highly optimized version of the code. The compiled code is then stored in a cache, ready to be executed when the spot is executed again. Note that as with a JIT, results of compilation are not kept between runs/users.

Static compiler

A static compiler compiles Java source code into the underlying machine’s native code that is then executed without interpretation. This approach is similar to traditional program development where the code is written, compiled and, if necessary, debugged.

Static compilation can also be applied to the bytecode generated by some Java compilers. This approach is applicable when the original Java source code is not available. Static compilation of Java bytecodes is possible because this code is the same across Java implementations and builds the basis for the portability of the Java language.

However, it is important to note that the portability of the Java application will not be affected by static compilation.

JVM tuning

Several tuning parameters of JVM could impact the performance of WebSphere Application Server and WebSphere Commerce Suite. They are:

- Heap size

The Java `mx` and `ms` parameter on the WebSphere Application Server are used to set the maximum heap for the JVM and the starting (minimum) heap for the JVM. The best performance will attain typically when `ms` and `mx` heap values are equal.

- JIT

The Just In Time (JIT) compiler is turned on by default on IBM JDK 1.1.8 for AIX and IBM JDK 1.1.7 for Windows NT, which results in better performance. For performance reasons turning off the JIT compiler does not help, but may be needed when debugging.

- Garbage collection

Turning off the garbage collection class enables more class reuse.

- Java stack and native stack size

Experiment with the value of the Java stack and stack size to optimize performance. Specifying both sizes of 819200 results in a slight performance improvement. If the values are set too low, this will result in performance degradation. If the values are set too high, this will result in a problem starting the application server.

5.3.1.2 HTTP Server

The HTTP or Web server is responsible for serving up everything from static pages to invoking various applications through interfaces such as Common Gateway Interface (CGI), FastCGI, Web server Application Programming Interfaces (APIs) and servlets. Due to the varied workload, careful planning is needed to ensure that the Web server is processing the requests in the most efficient manner.

Segmenting the work into similar functions is critical for producing consistent response times. Another technique to improve performance is to use the workload classification to make prioritized decisions on how to shed workload if the server becomes overloaded. Specific items in a Web server that impact performance are:

- Threads

Some Web servers allow the configuration of both a minimum and maximum number of threads. Pick a reasonable but large enough number of threads to handle your peak workloads. Then set the maximum number of threads equal to the minimum number. This avoids the overhead of creating and destroying threads during peak processing hours.

Experiment to find the right number of threads on your system. Experience has shown that picking too high a number can decrease your overall throughput on the system. Too low a number can even cause server failure. Remember in your calculations that the Web server uses some of the allocated threads for its own processing.

- Limit the use of server-side includes (SSI)

Server-side includes (SSI) allow you to insert information into CGI programs and HTML documents that the server sends to the client. When server-side include processing is enabled, the Web server will parse each byte of every HTML file and CGI program searching for the existence of an SSI directive and, if found, process it. This is a great feature for processing dynamic content, but it requires a large amount of CPU processing. SSI processing can be controlled by the use of the `imbeds` directive. If you do not use SSIs, set `imbeds off` in the `/etc/httpd.conf` file.

- URL links

URL suffix processing (multi-support) is overhead for any Web server. This occurs when a URL does not exactly match the templates on the `PASS` directive. For example, if the URL is `/oh_boy.html` and your file name is actually `/oh_boy.html.ascii`, the Web server will do suffix processing and eventually return the file `oh_boy.html.ascii`. The Web server appends all known suffixes to the file name looking for a match. Eventually, the correct file will be returned, if it exists, but the process is CPU intensive.

- Caching

Adding additional memory to a system almost always improves performance. This is because physical I/O is a relatively expensive operation in terms of latency. It makes intuitive sense then, that by dedicating memory in a Web server to store frequently accessed HTML pages and images, you will improve performance. As a rule of thumb, your Web server should have enough RAM to accommodate all network buffers, frequently used applications, images and HTML, including those mounted via DFS. This is especially important for dynamically generated pages that can be reused.

For the Web server there are several places you can cache your static items to help improve performance:

- Use a network router for caching
- Use a Web proxy cache such as the one found in WebSphere Edge Server.

- Logging

Web server logs, in particular the access log, record important information about the use of the Web server. However, these logs can become very large and should be pruned and/or archived regularly. Logging can have a surprisingly large impact on response time and throughput. For this reason, you will often find that vendors turn logging off for benchmarking purposes.

- CGI and server-side Java

The IBM Application Framework for e-business defines the infrastructure for developing e-business solutions. This includes the relationship between the Web server and server-side Java elements. IBM's analysis of the performance characteristics of this environment provides some useful guidelines for e-business solution designers, including:

- a. Any of the techniques used to connect a Web server to application logic (CGI, in-process API, Java Servlets, etc.) should be insignificant when compared to the time required to execute the application logic.
- b. Most existing Web applications have been implemented using CGI. IBM's analysis indicates that Java servlets provide 4X to 10X better throughput compared to CGI.
- c. Java Servlets generally run faster if they are instantiated and preloaded in *servlet.properties*. Servlet invocation by class name rather than instance name is slower.

- Network

If there is no contention for either CPU or memory resources on your system, and you are experiencing performance problems, you may have a network issue to resolve. After all, while incoming Web requests may be relatively small, outgoing Web responses can contain large graphics, applets, video, or audio files. It is important to make sure that the number and size of the TCP buffers be tuned appropriately on the Web server platform. The size of incoming requests may be very different from outgoing. Many sites have routed incoming requests along relatively "thin" network pipes while routing their output requests along relatively "fat" network pipes.

5.3.1.3 WebSphere Application Server

The WebSphere Application Server engine has parameters that affect the performance of the Web application. We will highlight a few of these parameters. For detailed information about WebSphere Application Server performance tuning, refer to the *WebSphere V3 Performance Tuning Guide*, SG24-5657 redbook.

Transport queue

The WebSphere Application Server is comprised of an enterprise bean container and servlet engine. A transport queue is used to route Servlet requests from the Web server to the servlet engine. Performance can be improved by adjusting the following transport queue parameters:

- Queue type

- Transport type
- Maximum connections

Servlets auto reload

WebSphere Application Server V3 has an auto reload capability. The auto reload specifies whether to automatically reload servlets in the Web application when the class files are changed. When changes are made to code, the application server does not need to be restarted, which simplifies testing and managing the Web site.

All that glitters is not gold. This feature has a negative impact on performance by dynamically reloading servlets and associated polling. It is advisable to turn auto reload off once the Web site is in production mode.

EJB container

The container cache parameters that could have an impact on performance are:

- Cache size
Cache size controls the number of buckets in the cache's hash table but not the size of the container cache.
- Cache preferred limit
The cache preferred limit is used by the cache manager as a trigger to start throwing unused entries out of the cache and is a "soft limit".
- Cache absolute limit
Provides an absolute limit of entries that will be maintained in cache by the container cache manager. The container will fail to allocate a new bean instance when the total number of active beans reaches this limit.
- Cache cleanup interval
Provides the interval in milliseconds for the background thread that attempts to ensure there are always free elements in cache. This parameter should be increased as the cache size increases in general.

Option A and option C caching performance consideration

The WebSphere Application Server provided with the WebSphere Commerce Suite provides Option A (exclusive) caching and Option C (shared) caching as defined by the EJB specification. With option C, the cache size has very little effect on performance, since beans are cached for the duration of the transaction that they are involved. With option A, there is a potential performance benefit from a large cache.

Number of containers

The best performance is attained by deploying all the EJBs in a single container per application server. When multiple containers are created, the processing overhead of JVM is increased without any gain in performance.

5.3.1.4 WebSphere Commerce Suite Server

This section provides performance considerations for the WebSphere Commerce Suite Server.

Page caching

As a rule of thumb, a page that is not cached takes .5 seconds to load versus 20 milliseconds for a cached page.

When a shopper clicks a link to view a product or category page only small amount of system time and resources are actually spent within the server. Most of the time is spent parsing the HTTP request, accessing the database, and dynamically creating the HTML page the shopper wants to see. Heavy site traffic and a large number of product and category entries in the database can further increase the time it takes for pages to load.

Most HTTP requests on the server will be for product and category pages, which are dynamically created by the ProductDisplay and CategoryDisplay commands, respectively. These commands retrieve information from your database, and display the information as an HTML page that has been generated from a Net.Data macro. If your product and category information have not changed since the page was last viewed, then it should not be necessary for the page to be dynamically re-created the next time a shopper requests it. Serving an equivalent “static” page that has been stored in a cache would be faster.

WebSphere Commerce Suite provides two methods for caching:

- Basic caching
- Advanced caching

Using one of these methods to cache your pages can ease the strain on the server and speed the download process for shoppers significantly.

The scope of the caching methods provided by Commerce Suite is mall wide. That means that if you have a WebSphere Commerce Suite instance that contains numerous stores, and you enable and configure caching, the caching will be turned on and work the same for all the stores in that mall (instance). Only the site administrator can configure and activate caching. Store administrators who only have store-level authority cannot.

In a multi-home configuration, a distinct storage location for cached files and distinct settings for the cache can exist for each instance of Commerce Suite. Files are stored in the cache on demand. This means that they are not stored by the file system until requested. Only those pages that are created by the commands `ProductDisplay` and `CategoryDisplay` are cached by the Commerce Suite caching methods. These commands query the `PRRFNBR` (product reference number) column in the `PRODUCT` table and the `CGRFNBR` (category reference number) column in the `CATEGORY` table, along with the merchant reference number. These name-value pairs are contained in the URL that creates the product or category pages for the shopper. If the file corresponding to the page being accessed is not in the cache file storage, it will be generated dynamically. This HTML page is then stored in the cache, and will not have to be regenerated until the data it is based on is modified.

If you change information in the database, such as a product price, you want to make sure that any page in the cache that contains the changed information will be deleted. Also, make sure that the page will be dynamically re-generated with the new information when the product is next viewed by a shopper. This is done for you by the synchronization daemon. When you change certain product or category information in the database, a log record is added to the `CACHLOG` table. The `PRODUCT` and `CATEGORY` tables populate the `CACHLOG` table using DB2 triggers. The synchronization daemon queries this table at regular, specified intervals, identifies which HTML pages are affected by the changes, and purges (deletes) them from the cache. This ensures that shoppers will not access any pages that are outdated or incomplete. In order for pages to be purged, the Synchronization Daemon must be running and the variable `NC_DMN_CACHE` must be active in the Commerce Suite configuration file.

Processes running

The WebSphere Commerce Suite Server process handles commands from the Web server. Each server process requires approximately 25 MB of memory. We suggest you start with four server processes per CPU and monitor CPU activity, paging and CPU wait activity. The number of server processes can be increased as long as the CPU activity is low, paging is very low and CPU wait is very low. The system should be tested during peak traffic loads. There is a point of diminishing returns when there too many server processes running; however, this point differs from application to application.

MS_TRANSCOUNT

The `MS_TRANSCOUNT` provides the server processes of WebSphere Commerce Suite the ability to recycle when reaching the value set in this parameter. The parameter can be found in the `ncommerce.ini/conf` file. The recycling is needed periodically to reclaim memory from memory leaks. If the

value is too small, there will be a lot of resources spent on stopping and starting jobs.

Logs

In production, the WebSphere Commerce Suite logs should be set to level 0 for performance consideration in order to minimize the amount of write activity. For example in log level 3, the highest level, WebSphere Commerce Suite server records the following information:

- Errors
- Status
- Debug
- HTML output

If the logging level is set higher than 0, the logs should be written to a separate disk to maximize parallel I/O activity.

Database size

Performance of an e-commerce application is affected by the size of the database. Sites that include large numbers of shoppers, merchants, or catalog items will typically not be able to serve requests as quickly as those that do not. Various factors will affect the throughput of your system, including database tuning, hardware capacity, using a well-designed schema, etc.

Page weight or MB/page

Page weight refers to the quantity of data that is returned from an average request. The suggested page weight is 9 KB/Page. If you are planning a site with significantly differing characteristics, be sure to compensate for your page size.

Note

A page includes requests for the initial URL, as well as all subsequent requests for images, etc.

Browser/buyer conversion rate

This is the ratio of requests for browsing traffic vs. requests for shopping traffic. A browser/buyer conversion rate of 95/5 is representative of most retail shopping sites. In general, buying commands have lower throughput than browsing commands. If the site has a significantly different browser/buyer conversion rate, be sure to compensate in your calculations.

Megabytes of data served per unit time

It is important to plan your site to have sufficient resources for serving expected network throughput. This includes sizing all firewall, routing, and other network components to be able to handle the peak workload expected of the site.

Also, if your site will be running webcasts or real-time audio this will demand bandwidth. Although this is outside the scope of the commerce engine and must be accounted for in the overall solution.

5.3.2 Database server

The WebSphere Commerce Suite operates as an Online Transaction Processing (OLTP) application. The database used in this environment should then be configured and tuned to support a high level of traffic and provide quick responses.

It is always the best practice to store the database on a separate machine from the WebSphere Commerce Suite Server. The 2-tier or 3-tier runtime topologies are the most scalable architectures.

5.3.2.1 General DB2 performance guidelines

The following issues should be taken into consideration when tuning a DB2 server:

- CPUs and memory

There should be a minimum of 2 CPUs on the database server and there should be 512 MB RAM available for each processor.

- Storage devices

Operating system (O/S), O/S swap, database and database logs should be located on separate disks to take advantage of parallel I/O processes. The O/S swap should be set to 2 or 2.5 times to the physical memory on the machine. Also, we suggest that you use *off-by-n* mirroring for the database disks, that is, use each disk for both data and mirroring by splitting the disk in half and putting data in one half and a mirror on the other.

Note

Do not use Raid 5 for drives where you store databases, for performance reasons.

- Tablespaces, containers and bufferpools

Use the default tablespaces

- Monitoring
 - Paging space incurring

There should be no paging for the pages paged in and out. On AIX, use the `vmstat` command to monitor paging. If it occurs, then add more memory to the system. The lower CPU usage the better. We recommend you keep CPU utilization less than 85%.
 - CPU usage on I/O process

It is important to monitor the relationship of CPU and I/O process. On AIX, use the `iostat` command on the database server. The lower the CPU usage the better. Take note of the amount of time for the CPU(s) are waiting idle on an outstanding disk I/O request. This should be less than 5%. Also the I/O should be even across the disks.
 - Even monitoring

Use the event monitoring facilities to capture SQL statements. Execute the `explain` command to identify and correct expensive SQL.
- Use the Database Cleanup utility to remove obsolete data.
- Optimize the DB2 database
 - Use the DB2 `reorgchk` utility on tables that are frequently updated.
 - Use the DB2 `runstats` utility on tables that are frequently updated.
- Create additional indexes

Create additional indexes to reduce the time it takes to locate rows in tables. The database uses indexes for its most frequently accessed columns. You can create an index for any column that is not listed. Note that while an index may improve performance, it also takes up more space in the database.
- Optimize SQL queries
 - Increase the size of the buffer pool
 - Use the DB2 `runstats` utility
 - Create indices to columns referenced most often by SQL queries
- Increase the size of the buffer pool

The Database Manager uses the buffer pool to cache data and index pages. Storing data in the buffer pool reduces I/O usage. Increasing the buffer pool size gives the Database Manager more memory to load database indexes. If possible you would want to load the entire database into memory. To increase the size of the buffer pool use the commands `ALTER BUFFERPOOL` or `UPDATE DATABASE CONFIGURATION`.

- Set up table space on multiple hard drives

To customize table space, do the following:

- a. Edit the following file:

Windows NT:

```
\\IBM\CommerceSuite\nc_schema\db2\customized_userspace.nt.db2.sql
```

AIX:

```
/usr/lpp/CommerceSuite/customized_userspace.aix.db2.sql
```

- b. From a DB2 command window run the script.

5.3.2.2 WebSphere Commerce Suite DB2 performance guidelines

- Load a specific WebSphere Commerce Suite database table to memory, for example, the PRODUCT table
- Run command to let the database system to collect statistics on the data and reorganize the database itself.
- Run the ncclean command provided with the WebSphere Commerce Suite to clean up the database.

5.3.2.3 Oracle

Most of the items done for DB2 can be applied to Oracle such as optimizing the SQL queries, increasing the size of the buffer pool, creating additional indices, etc.

For more information, refer to Oracle's documentation.

5.3.3 Network Dispatcher

The need to manage and scale Web sites that experience high *hit rates* over the Internet has led to the development of load balancing application server designs. Load balancing designs range from the relatively simple round-robin Domain Name System (DNS) approaches to more sophisticated dispatchers. The most common approach to load balancing today is Web server clustering. A clustered environment in the Web world is a set of Web servers that appear to browsers on the Internet as a single server. A required element in a clustered environment is a load balancer, which is software or hardware that is responsible for making the set of Web servers appear to be a single server. For example, the SecureWay Network Dispatcher component of IBM's WebSphere Performance Pack provides this function.

5.3.3.1 Technology considerations

The Network Dispatcher creates the illusion of having just one server by grouping systems together into a cluster that behaves as a single, virtual server. The service provided is no longer tied to a specific server system, so you can add or remove systems from the cluster, or shut down systems for maintenance, while maintaining continuous service for your clients.

For Web clients, the balanced traffic among servers seems to be a single, virtual server, and the site appears as a single IP address to the world. All requests are sent to the IP address of the dispatcher machine. The dispatcher decides for each client request which server is the best one to accept requests, according to certain dynamically set weights. The dispatcher routes the client's request to the selected server, and then the server responds directly to the client without any further involvement of the dispatcher.

The dispatcher can also detect a failed server and route traffic around it. The dispatcher receives the packets sent to the cluster. These packets have a source and a destination address, the destination address is the IP address of the cluster. All servers in the cluster and in the dispatcher system have their own IP address and an alias for the IP address of the cluster. The dispatcher system has the cluster address aliased on the network interface, while all the TCP servers that will be load balanced by this Network Dispatcher (ND) machine have the cluster address aliased on the loopback adapter. The dispatcher system checks which server is the next best server to handle the load and routes the packet to that server. The dispatcher routes this request based on the hardware address of the network adapter (MAC address) of the chosen server. It changes the hardware address of the packet to the hardware address of the selected server and sends the packet to the server. However, the dispatcher does not change the source and destination IP addresses in the packet. The server receives the packet and accepts it because all servers in the cluster have an alias for the cluster's IP address on the loopback interface. Then, the server sends a response back to the client by inverting the source and destination IP addresses from the original packet received. This way, the server can respond directly to the client.

The fact that the server can respond directly to the client makes it possible to have a small-bandwidth network for incoming traffic, such as Ethernet or token-ring, and a large-bandwidth network for outgoing traffic, such as Asynchronous Transfer Mode (ATM) or Fiber Distributed Data Interface (FDDI).

5.3.3.2 High availability

The Network Dispatcher has a high availability feature. It involves the use of a secondary machine that monitors the main, or primary, machine and stands by to take over the task of load balancing, should the primary machine fail at any time.

In case of failure, clients lose only the current connections, but they can immediately establish a new connection to the remaining servers with no problems. The high-availability environment involves two dispatcher machines with connectivity to the same clients, and to the same cluster of servers, as well as connectivity between the dispatchers. Both the dispatchers must be using the same operating systems. The two dispatcher machines are usually referred to as primary machine and backup machine:

- The primary machine works normally as a dispatcher, and is in the active state while it is balancing the load among the servers of its clusters.
- The backup machine, configured in a very similar way to the primary machine, stays in standby mode unless the primary fails.

The two machines are synchronized, and only the primary machine routes packets, while the backup machine is continually updated. The two machines establish communication to monitor the status of each other, referred to as a heartbeat, using a port that you can choose. If the primary machine fails, the backup machine detects this failure, switches to active state, and begins to take over the routing of packets. When the primary machine is operational again, but in standby state, you can either decide that it again automatically becomes the active machine, or leave it in standby mode. In this case, you will have to act manually if you want it to become the active machine again.

5.3.4 Integration to back-end system

Integration of Web application servers with back-end systems also raises several performance issues, including:

1. Pre-allocating and caching resource manager

Some of the cost associated with accessing resource managers from middle-tier objects involves establishing connections to those resource managers. By pre-allocating and reusing connections, it is possible to greatly reduce the overhead of defining new ones.

2. Caching facilities local to the middle tier

Caching state information accessed from back-end systems into the middle-tier environment can promote good performance. This is achievable through pre-fetch and look-aside algorithms that ensure that

the back-end will only be accessed as often as is absolutely necessary. Note that it is common, especially when integrating with existing legacy information systems, to access back-end state information required by one object and, in the process, to encounter state information needed by one or more other, related objects. Caching this information as it becomes available may significantly improve overall system performance. Finally, a smart caching mechanism can ensure that updates are made to a back-end system only when the underlying state information of a given object has actually changed.

3. Optimistic locking mechanisms

In some cases, applications place large numbers of unnecessary locks on resource managers. We say these locks may be unnecessary in the sense that no attempts to use the resource concurrently actually occur. In these situations it would be better to place no locks on the resource managers, and instead check for conflicting usage as part of transactional commit. Specifically, you should consider locking all affected resources only once: at the end of a given unit of work. Then compare the relevant resource manager values at that time with the values obtained when the unit of work began. In the absence of intervening changes, the current unit of work can be committed. Otherwise, it can be rolled back with an exception returned to the client. In either case, you may be able to reduce overall locking overhead and improve total system throughput and performance. Note that this “optimistic” locking strategy is inappropriate for some types of applications and domains. When updates to the same resource occur on a frequent basis, traditional or *pessimistic* locking mechanisms should be used. It’s also worth mentioning that multiple resources should typically be accessed by multiple applications in the same sequence to reduce potential *deadlock* situations.

4. Implementation pushdown

Object-based solutions that must coexist with legacy systems should complement (versus compete with) these systems. For example, a query invoked on a virtual collection of objects whose state maps to a relational database manager should first be translated into native SQL statements. These SQL statements, processed using the optimization technology provided by the database manager, can then return a result set whose values implicitly identify candidate objects satisfying the query. In this way, a minimal amount of processing is required in “object space”, leaving the bulk of the work to be handled instead by resource managers that have been perfecting high-performance solutions over the course of many years. As a final point that is specific to our query example, it’s notable that an object-based implementation of query should also be able to return

multiple elements back from a single method call, and that the query result set should be demand-driven (meaning that objects should only be activated on the server if a client actually requests them).

5.4 Application development

In WebSphere Commerce Suite Pro Edition, the default technology to generate most pages is Net.Data, database queries and formatting the results using Net.Data macros and templates or JSPs. Typical stores also contain some static content, such as HTML page requests.

In this section, we will discuss the performance consideration of the following application development technologies to be used typically in a WebSphere Commerce Suite Web site:

- Net.Data
- HTML
- JSPs and Servlets

5.4.1 Net.Data

Consider the following issues to increase the performance of Net.Data macros used within WebSphere Commerce Suite:

- Members of DB2WWW.ini for the commerce instance contain paths that are used by WebSphere Commerce Suite to locate macros and the includes for those macros. Invoking a macro or calling an include file inside a Net.Data macro will cause WebSphere Commerce Suite to search through the list of paths in the MACRO_PATH or INCLUDE_PATH. If these variables contain many entries, a great deal of searching will be done. We recommend that you decrease the number of entries by putting all macros in a single directory, if possible.
- Avoid writing very large macro and include files if only a small portion is executed most times inside the macro. If it is a large macro and include file, Net.Data has to parse the entire file with the included files expanded regardless of how much of the macro is executed. It is better to write smaller, more specialized include files that get used where they are needed.
- Do not include table columns that are not needed by the Net.Data macro. Remove unnecessary white space from macros. This will speed up transmission of the file but makes the macro less readable.

5.4.2 HTML pages

Static page serving

Static pages are served at roughly the same rate as static images. Requests for static pages can be ignored in capacity planning if the ratio of dynamic requests to static requests you are expecting at your site is fairly high. For example, a site that serves two dynamic pages for every static page served can generally be sized without regard for the static pages, since dynamic pages require substantially more resources to serve.

In HTTP 1.1, a static page is connectionless. Although it is connectionless, the protocol remains active to ensure all packets were sent successfully by the works of TCP.

HTTP is asymmetric. It means the number of request bytes from the client is much smaller than the reply bytes from the server. If the total request packet and sent packet most likely have the same amount, then the Web server becomes a bottleneck.

We recommend that you keep the size of the static page smaller or equal to the router packet size. If the size of a Web page is bigger than the maximum router packet size, the router will split the packet to its maximum capacity. For example if your router packet size is 1500 KB and your Web page is 2000 KB the router will split the packet into one 1500 KB packet and one 500 KB packet. In this example, the 500 KB is using 33% of the capacity of the router packet size.

Dynamic page serving

Normally, dynamic Web pages require script programming such as Net.Data. Dynamic Web pages produces the HTML content sent to the Web browser. It uses HTML for formatting the dynamic information, such as information retrieved from a database. Interactive HTTP sessions generate lots of interruptions that affect the performance of the Web server. Every interactive movement is sent back to the server and creates more traffic on the network. We recommend the use of the page caching facilities from both the Web server and WebSphere Commerce Suite to reduce the traffic.

5.4.3 Servlets and JSPs

We will discuss the performance considerations of the non-persistent servlet model, the persistent servlet model, and JSPs.

5.4.3.1 Servlets non-state aware model

This model is similar to the CGI environment. It requires that a user enter information into an HTML form of a Web browser-based application and

submit the form. After the response is sent back to the Web browser, the transaction is finished.

Browser considerations

In this environment, an HTML document is downloaded from the HTTP server to the Web browser. The document uses some HTML tags to enable user input. From the performance perspective, non-state aware HTML is the same as any static HTML. The HTML document and images are all separate objects needing to be served.

Server considerations

Compared with the limitations of the CGI environment, servlets are more efficient for enabling dynamic Web applications. Servlets run under the HTTP server process and do not require the extra workload of additional job creation. Servlets also access the system services directly. Servlets also remain resident after being initially loaded, and can be re-used. This helps to provide a 4 to 10 times better performance than CGI programs. It also helps to minimize the time needed to connect to the actual application logic.

5.4.3.2 Servlet persistent model

In the WebSphere Commerce Suite, servlets are implemented in a transaction-oriented environment. For example, servlets can be used to keep track of the products in the shopping cart.

Browser considerations

In this environment, you might use HTTP *cookies* to maintain a state. A cookie is a local text file that contains information about your Web browser session for a Web site. Cookies must be enabled on the browser or persistence cannot take place. When using cookies, the performance penalty is negligible for the Web browser client.

Server considerations

The server application is responsible for transaction management tasks. The application must be prepared to accept multiple service requests. Session management is necessary for the transaction and is part of the application.

5.4.3.3 JSPs

JSPs and Servlets can be used separately or together. A common implementation strategy uses Servlets for transaction-oriented tasks. The servlet places its result in HTML for the end user, or in a JavaBean bean component. A JSP can retrieve information from this “transaction” and then provide appropriate user interface to the browser.

Since JSPs are essentially dynamically created Servlets, you may refer to the topics covered in, “Server considerations” on page 74. Since the Java code contained in a JSP must be compiled then executed, the more Java code include, the longer response time and the greater the server load. JSPs can run in a dynamic, interpreted mode that requires no configuration on the server.

If using JavaBeans in the JSP, the Beans.instantiate() method looks first for a serialized instance (.ser file). If this is not found, a new newInstance() function is called, which also impacts performance.

5.5 General performance guidelines

When tuning servers or technologies for performance, remember to consider the following:

- Pay attention to all components of the solution.
- Understand in detail the interfaces and flows between the various components. For example, tuning your HTTP server may result in exposing a performance bottleneck in your commerce or database server.
- Plan for growth in the design.
- Use the latest levels of infrastructure and system software.
- Cache as much as possible.

5.6 Where to find more information

IBM Publications:

- *WebSphere V3 Performance Tuning Guide*
- *WebSphere Scalability: WLM and Clustering Using WebSphere Application Server Advanced Edition*
- *IBM WebSphere Performance Pack: Load Balancing with IBM SecureWay Network Dispatcher*
- Maggie Archibald, Mike Schlosser: *Designing e-business Solutions for Performance* white paper at:
<http://www.ibm.com/software/developer/library/patterns/performance.html>
- Documentation for IBM WebSphere Application Server, including the product library (which includes IBM HTTP Server documentation), hints and tips, white papers, and other support information can be found at:
<http://www.ibm.com/software/webservers/appserv/support.html>
- Information on IBM WebSphere Application Server, specifically for AS/400, including documentation, support, and performance considerations, can be found at: <http://www.as400.ibm.com/websphere/>

Other Publications:

- JavaSoft: *The Java HotSpot Performance Engine Architecture* white paper at: <http://java.sun.com/products/hotspot/whitepaper.html/>.

Chapter 6. Technology options

This chapter looks at the technologies that you should consider when developing e-commerce and Web applications. The technologies are based upon the open standards and Java-based programming model of the IBM Application Framework for e-business. The WebSphere Commerce Suite and the WebSphere Application Server are Web application servers that are based upon the open standards and Java-based programming model of the IBM Application Framework for e-business.

Specifically, we will provide guidelines for the technology options available within each of the following:

- Web clients
- WebSphere Application Server
- WebSphere Commerce Suite
- Where to find more information

6.1 Web clients

There are two types of Web clients, the application client and the Web browser client.

- Application client

Application clients are primarily large Java applets or Java applications. These clients provide rich graphical user interfaces compared to HTML clients. They may communicate with the Web application server over a number of protocols including HTTP, IIOP, MQ, etc. Application clients communicate with the Web application server primarily to receive data rather than pre-formatted HTML pages. All of the user interface processing is performed on the client side. In addition, under this model, some parts of the business logic can also be processed on the client side. These kind of client applications are not covered in this book.

- Web browser client

A Web browser client is an application client that uses a Web browser such as Netscape Navigator or Microsoft Internet Explorer. Web browser clients use the HTTP protocol to communicate with the Web application server to display HTML. In addition, they are capable of processing client-side JavaScript for enhancing navigation to perform simple input validation and to handle simple errors. The Application Framework recommends “thin clients” with little or no application logic. Applications

are managed on the server and downloaded to the requesting clients. The client portions of the applications should be implemented in HTML, dynamic HTML (DHTML), XML, and Java applets as seen in Figure 14 on page 78.

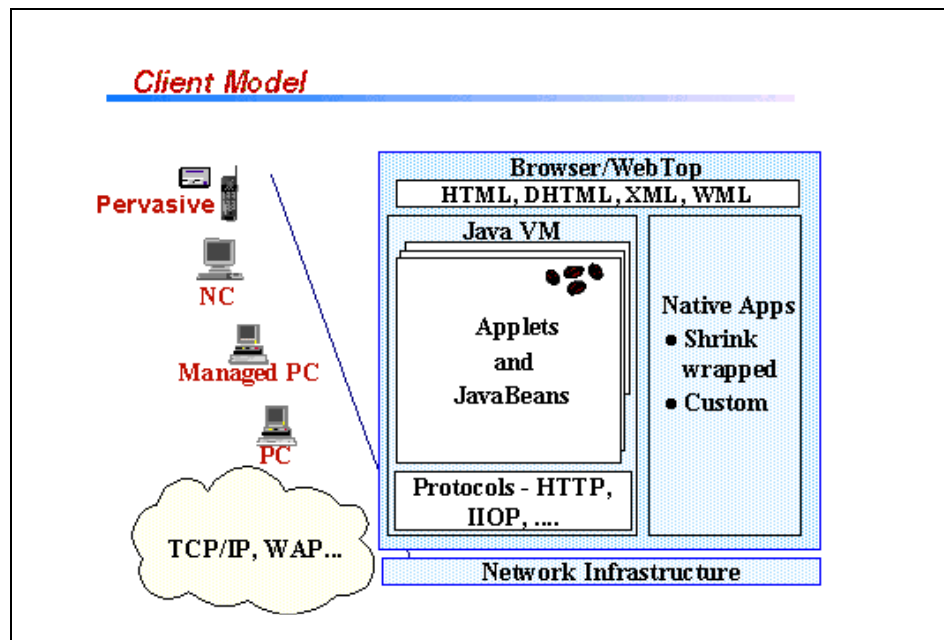


Figure 14. Web client technology model

The following sections outline some of the possible technologies that should be considered, but remember your choices may be constrained by the policy of your customer. For example, for security reasons, only HTML is allowed on browser clients at some government agencies.

6.1.1 Web browser

A Web browser is a fundamental component of the Web client. For PC-based clients, the browser typically incorporates support for HTML, DHTML, JavaScript and Java. Some browsers are beginning to add support for XML as well. Under user control, there is a whole range of additional technologies that can be configured as "plug-ins", such as RealPlayer from RealNetworks or Macromedia Flash.

As an application designer you must consider the level of technology you can assume will be available in the user's browser. You can add logic to your

application to enable slight modifications based upon the browser level. Also, when adding features that require a plug-in, you need to consider what portion of your intended user community will have that capability.

For an e-business application that is to be accessed by the broadest set of users with varying browser capabilities, the client is often written in HTML with no other technologies. On an exception basis, limited use of other technologies, such as using JavaScript for simple edit checks, can then be considered based on the value to the user and the policy of the organization for whom the project is being developed.

The emergence of pervasive devices introduces new considerations to your design with regard to the content streams that the device can render and the more limited capabilities of the browser. For example, WAP (Wireless Application Protocol) enabled devices render content sent in WML (Wireless Markup Language).

6.1.2 Markup languages

Markup languages use a set of labels that are embedded within text to distinguish individual elements or groups of elements for display or identification purposes. The labels are typically known as *tags*. Markup languages identify elements within a continuous stream of text rather than more structured data in a database. However, XML is a markup language that turns text streams into the equivalent of database records. We will review the following markup languages used on the Web client:

- HTML
- Dynamic HTML
- XML, XSL

6.1.2.1 HTML

HTML is a document markup language with support for hyperlinks, that is rendered by the browser. It includes tags for simple form controls. Many e-business applications are assembled strictly using HTML. This has the advantage that the client-side Web application can be a simple HTML browser, enabling a less-capable client to execute an e-business application.

The HTML specification defines user interface (UI) elements for text with various fonts and colors, lists, tables, images, and forms (text fields, buttons, checkbooks, and radio buttons). These elements are adequate to display the user interface for most applications. The disadvantage, however, is that these elements have a generic look and feel, and they lack customization. As a result, some e-business application developers augment HTML with other

user interface technologies to enhance the visual experience, subject to maintaining access by the intended user base and compliance with company policy on Web client technologies.

Almost all browsers support HTML V3.2, which is often the lowest common denominator for building the client side of an application.

6.1.2.2 Dynamic HTML (DHTML)

Dynamic HTML (DHTML) allows a high degree of flexibility in designing and displaying a user interface. In particular, DHTML includes cascading style sheets (CSS) that enable different fonts, margins, and line spacing for various parts of the display to be created. These elements can be accurately positioned using absolute coordinates.

Another advantage of DHTML is that it increases the level of functionality of an HTML page through a document object model and event model. The document object enables scripting languages such as JavaScript to control parts of the HTML page. For example, text and images can be moved about the screen, and hidden or shown, under the command of a script. Also, scripting can be used to change the color or image of a link when the mouse is moved over it, or to validate a text input field of a form without having to send it to the server.

Unfortunately there are several disadvantages with using DHTML. The greatest of these is that two different implementations (Netscape and Microsoft) exist and are found only on the more recent browser versions. A small, basic set of functionality is common to both, but differences appear in most areas. The significant difference is that Microsoft allows the content of the HTML page to be modified by using either JScript or VBScript, while Netscape only allows the content to be manipulated (moved, hidden, shown) using JavaScript.

Due to the browser incompatibility issues, DHTML is not recommended in environments where mixed levels and brands of browsers are present.

6.1.2.3 XML, XSL

Extensible Markup Language (XML) allows you to specify your own markup language with tags specified in a Document Type Definition (DTD). Actual content streams are then produced that use this markup. The content streams can be transformed to other content streams by using XSL (eXtensible Stylesheet Language).

XML is a framework for defining document markup languages and is predicted to become the primary approach to document exchange over the

Internet. In simple terms, a document markup language is a set of elements (frequently called tags) that have one or more of the following functions:

- Describe the structure of the document
- Describe the content of the document
- Control how the document is presented to the user

XML and Hypertext Markup Language (HTML) are derived from the more complex Standard Generalized Markup Language (SGML). SGML's complexity and high cost of implementation spurred the interest in developing alternatives.

HTML is the most widely used markup language for Web documents. As the popularity of HTML increases, the limitations of the language have become more apparent. The limitations of HTML include:

- Restricting the user to a relatively small set of tags
- HTML authors cannot create their own HTML tags. Commercially available Web browsers have no knowledge of tags that are not part of the HTML standards.
- Control presentation is contained in the same file as the tags that describe the document content.
- Although HTML 4 and cascading style sheets enable HTML authors to separate content from presentation, HTML 4 remains weak in its ability to describe the content of a document.

XML overcomes many of the limitations of HTML and other markup languages, while providing capabilities that are not a part of the earlier languages. In the XML document, the tag names convey the meaning of the data they contain. The structure of the document is easily discerned and follows a pattern. In contrast, the HTML tag names reveal little about the meaning of their content and the structure is not particularly useful for manipulating the document and exchanging it between applications.

XSL is a language for expressing style sheets and provides two major functions with XML:

- Language for transforming XML documents
- XML vocabulary for specifying formatting semantics

An XSL style sheet specifies the presentation of a class of XML documents by describing how an instance of the class is transformed into an XML document that uses the formatting vocabulary.

6.1.3 Client-side scripts

JavaScript is a cross-platform object-oriented scripting language. JavaScript is a great utility language for Web applications because of the browser and document objects that the language supports. Client-side JavaScript provides the capability to interact with HTML forms. You can use JavaScript to validate user input on the client and help improve the performance of your Web application by reducing the number of requests that flow over the network to the server.

ECMA, a European standards body, has published a standard (ECMA-262) that is based on JavaScript (from Netscape) and JScript (from Microsoft) called ECMAScript. The ECMAScript standard defines a core set of objects for scripting in Web browsers. JavaScript and JScript implement a superset of ECMAScript.

To address various client-side requirements, Netscape and Microsoft have extended their implementations of JavaScript in Version 1.2 by adding new browser objects. Due to Netscape and Microsoft extensions beyond the standard JavaScript 1.2, the extensions must detect the browser being used and select the correct statements to run.

The use of JavaScript on the server side of a Web application is not recommended, given the alternatives available with Java. Where your design indicates the value of using JavaScript, for example for simple edit checking, use JavaScript 1.1, which contains the core elements of the ECMAScript standard.

JavaScript: The Definitive Guide, Third Edition, by David Flanagan, is an excellent book on JavaScript that details the JavaScript objects and methods listing their origin and JavaScript level.

6.1.4 Java applets

Java applets provide the most flexible user interface (UI) technology that can be run in a Web browser. Java provide a rich set of UI elements in comparison to the equivalent HTML UI elements. In addition, Java's rich programming language provides an infinite set of UI elements that can be built and used. There are many libraries available that offer common UI elements, such as tables, scrolling text, spreadsheets, editors, graphs, charts, etc.

A Java applet is a program written in Java that is downloaded from the Web server and run on the Web browser. The applet to be run is specified in the HTML page using an APPLET tag:

```
<APPLET CODEBASE="/mydir" CODE="myapplet.class" width=400 height=100>  
  <PARAM NAME="myParameter" VALUE="myValue">  
</APPLET>
```

In this example, a Java applet called myapplet will run. Parameters provide an effective way to send data to an applet with the use of the PARAM tag. The applet has access to the parameter data and can easily use it as input to the display logic.

Java can also request a new HTML page from the Web application server. This provides an equivalent function to the HTML FORM submit function. The advantage of loading a new HTML page with an applet is that it can do the obvious (a button being clicked) or the unique (the editing of a cell in a spreadsheet), unlike HTML forms.

Java applets seldom consist of just one class file. Large applets may reference hundreds of class files. Making a request for each of these class files individually can tax any server and also tax the network capacity. To address this issue, Java provides a means to package the class files into a JAR or CAB file. This reduces the number of load requests from hundreds to just one. Netscape and HotJava support JAR files simply by adding an ARCHIVE="myjarfile.jar" variable within the APPLETTAG tag. Internet Explorer uses CAB files specified as an applet parameter within the APPLETTAG tag. In both cases, executing an applet contained within a JAR or CAB file deliver faster load times than individual class files. While Netscape and Internet Explorer use different APPLETTAG tags to identify the packaged class files, a single HTML page containing both tags can be created to support both browsers. Each browser simply ignores the other's tag.

A disadvantage of using Java applets for UI generation is that it requires a version of Java that is supported by the Web browser. Often, the Java Virtual Machine (JVM) contained within the browser lags behind the most current and capable versions of the Java Developers Kit (JDK), which presents a dilemma. Do I force my user community to use a specific version of a browser, or do I use an older version of the Java technology that is supported within the Web browsers for the client-side application?

Note

The leading browsers from Netscape and Microsoft support variant levels of JDK 1.1 and each has different security models for signed applets.

Another disadvantage of Java applets is that any class such as widgets, and business logic that are not included as part of the Java support in the browser must be loaded from the Web server as they are needed. If these additional classes are large, the initialization of the applet may take from seconds to minutes, depending upon the speed of the connection to the internet.

Due to the shortcomings discussed, the use of Java applets is not recommended in environments where mixed levels and brands of browsers are present. Small applets may be used in rare cases where HTML UI elements are insufficient to express the semantics of the client-side Web application user interface. If it is absolutely necessary to use an applet, care should be taken to include UI elements that are core Java classes whenever possible.

6.2 WebSphere Application Server

This section is focused on Java server-side programming for the Web application server. The Application Framework for e-business recommends the technology model seen in Figure 15 for a Web application server.

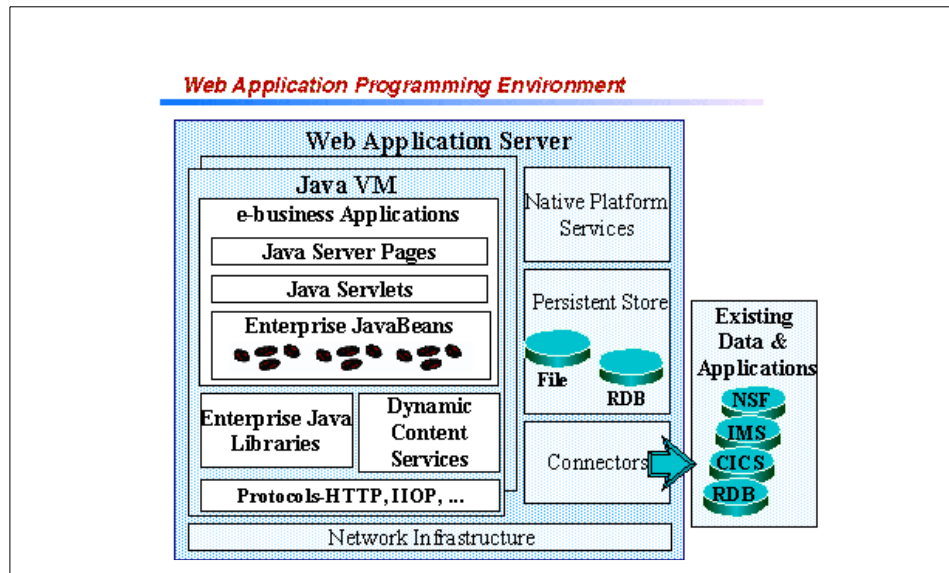


Figure 15. Web application programming environment

While there have been many other models for a Web application server, the Framework has had great industry support. For more details on the Java APIs discussed in this section, see *Java Enterprise in a Nutshell*.

Before we look at the technologies and APIs available in the Web application programming environment, we need to review the operational components on this node. The selection of the Web/HTTP server and the Java Virtual Machine (JVM) are critical to the application in areas such as robustness, performance and availability.

In Chapter 7, “Application design guidelines” on page 105, we discuss the Model-View-Controller (MVC) design structure for user interfaces. In summary, the MVC for the Web application programming model states the following:

- Model
The model is represented to the view and interaction controller by using a set of JavaBean components.
- View
The view is generally best implemented using JSPs.
- Controller
The interaction controller, which is primarily concerned with processing the HTTP request and invoking the correct business or UI logic, often lends itself to an implementation as a servlet.

6.2.1 XML

XML and XSL style sheets can be used on the server side to encode content streams and parse them for different clients, thus enabling you to develop applications for both a range of PC browsers and for the emerging pervasive devices. The content is in XML format and an XML parser is used to transform the XML to output streams presented within XSL style sheets. This general capability is known as transcoding and is not limited to XML-based technology.

The appropriate design decision here is how much control over the content transformation you need in your application. You will want to consider when it is appropriate to use this dynamic content generation and when there are advantages to having servlets or JSPs specific to certain device types.

XML is also used as a means to specify the content of messages between servers, whether the two servers are within an enterprise or represent a business-to-business connection. The critical factor here is the agreement between parties on the message schema, which is specified in an XML DTD document.

An XML parser is used to extract specific content from the message stream. Your design will need to consider whether to use an event-based approach, for which the SAX API is appropriate, or to navigate the tree structure of the document using the DOM API. For more information visit:
<http://www.sun.com/xml/>.

6.2.2 JavaServer Page (JSP)

JavaServer Pages (JSP) were designed to simplify the process of creating pages by separating Web presentation from Web content. In the page construction logic of a Web application, the response sent to the client is often a combination of template data and dynamically generated data. In this situation, it is much easier to work with JSPs than to do everything with servlets.

The chief advantage JSPs have over Java servlets is that they are closer to the presentation medium. A JSP is an HTML page. JSPs can contain all the HTML tags that Web authors are familiar with. A JSP may contain fragments of Java code that encapsulate the logic that generates the content for the page. These code fragments may call out to beans to access reusable components and back-end data. JSPs are compiled into servlets before being executed on the Web application server. JSPs are the recommended choice for implementing the “View” for the Web browser client. For those cases where the code required is a large percentage of the page content, and the HTML minimal, writing a Java servlet is a better choice, due to readability and maintenance reasons.

The current level of the JSP API is 1.1. To learn more about JSPs visit:
<http://www.javasoft.com/products/jsp/>.

6.2.3 Java Servlets

Java Servlets or *servlets* provide a replacement for CGI-based techniques in Web application programming. Servlets provide a component-based, platform-independent method for building Web-based applications, without the performance limitations of a CGI program. Servlets interact with the servlet engine on the Web server, through HTTP requests and responses that are encapsulated as objects in the servlet.

One of the attractions of using servlets is that the API is very accessible for a Java programmer to master. Servlets are a core technology in the Web application programming model. They are the recommended choice for implementing the “Interaction Controller” classes that handle HTTP requests received from the Web client.

The current level of the servlet API is 2.2. To learn more about Java servlets visit: <http://www.javasoft.com/products/servlet/>.

6.2.4 JavaBeans

The JavaBeans or *beans* component architecture enables developers to create reusable software components that can then be assembled together using visual application builder tools, such as IBM VisualAge for Java or IBM WebSphere Studio. JavaBeans also known as “beans” may be visual or non-visual.

Recommended uses of beans with JSPs and servlets:

- As the client interface to the “Model Layer”. An “Interaction Controller” servlet will use the bean interface.
- As the client interface to other resources. In some cases this may be generated for you by a tool.
- As a component that incorporates a number of property-value pairs for use by other components or classes. For example, the JSPs specification includes a set of tags for accessing JavaBean properties.

The current level of the JavaBeans API is 1.01. To learn more about JavaBeans visit: <http://www.javasoft.com/products/beans/>.

6.2.5 Enterprise JavaBeans (EJB)

Enterprise JavaBeans (EJB) are distinguished from JavaBeans in that they are designed to be installed on a server, and accessed remotely by a client. The EJB framework provides a standard for server-side components with transactional characteristics.

The EJB framework specifies the responsibilities of the EJB developer and the EJB container provider. The intent is that the “plumbing” required to implement transactions or database access can be implemented by the EJB container. The EJB developer specifies the required transactional and security characteristics of an EJB in a deployment descriptor (this is sometimes referred to as declarative programming). In a separate step, the EJB is then deployed to the EJB container provided by the application server vendor of choice.

There are two types of Enterprise JavaBeans:

1. Session
2. Entity

Session bean characteristics:

- Executes on behalf of a single client
- Can be transactional
- Can update data in an underlying database
- Is relatively short lived
- Is destroyed when the EJB server is stopped. The client has to establish a new session bean to continue computation.
- Does not represent persistent data that should be stored in a database.
- Provides a scalable runtime environment to execute a large number of session beans concurrently.

Entity bean characteristics:

- Represents data in a database
- Can be transactional
- Shared access from multiple users
- Can be long lived (lives as long as the data in the database)
- Survives restarts of the EJB server. A restart is transparent to the client.
- Provides a scalable runtime environment for a large number of concurrently active entity objects.

Typically an entity bean is used for information that has to survive system restarts, while in session beans, the data is transient and does not survive when the client's browser is closed. For example, a shopping cart containing information that may be discarded uses a session bean, and an invoice issued after the purchase of the items is an entity bean.

An important design choice when implementing entity beans is whether to use Bean Managed Persistence (BMP), in which case you must code the JDBC logic, or Container Managed Persistence (CMP), where the database access logic is handled by the EJB container.

The business logic of a Web application often accesses data in a database. EJB entity beans provide a convenient way to wrap the relational database layer in an object layer, hiding the complexity of database access. A single business task may involve accessing several tables in a database. Modeling rows in those tables with entity beans makes it easier for your application logic to manipulate the data.

The current level of the Enterprise JavaBean API is 1.1. The most significant changes from EJB 1.0 are the use of XML-based deployment descriptors and the need for vendors to implement entity bean support to claim EJB compliance. To learn more about Enterprise JavaBeans visit:
<http://www.javasoft.com/products/ejb/>

6.3 WebSphere Commerce Suite

To build and run your e-commerce site, you need software that provides a secure, scalable and dependable environment that you can trust. WebSphere Commerce Suite provides you with software that has powers some of the busiest, most successful e-commerce sites available online. The WebSphere Commerce Suite is packaged with the following award-winning software to provide a complete end-to-end solution:

- IBM DB2 Universal Database provides a scalable, reliable database to store your data.
- IBM HTTP Server provides a powerful Web server and is based on the popular Apache Web server.
- IBM WebSphere Payment Manager provides a secure system for managing payments.
- IBM WebSphere Application Server for servlets and JSP technology.
- IBM SecureWay Directory is a robust directory server.
- Blaze Rule Server provides rules processing for personalized product recommendations WebSphere Commerce Suite can also be integrated with other software products.

For more information about other runtime components, refer to the WebSphere Commerce Suite, *Installation Guide*.

To meet your core business needs, the WebSphere Commerce Suite helps you in the following key areas:

- Content Management
Allows you to manage your catalog content to keep your site up-to-date and interesting.
- Marketing
Helps you present valuable information to your customers and execute marketing strategies to increase sales.
- Order Management

Allows you to manage and understand your store's sales.

- Fulfillment

Allows you to manage product delivery.

- Customer Management

Provides customers with superior services and support through order and shipment tracking.

- Payments

Enables secure electronic transactions.

WebSphere Commerce Suite exploits the WebSphere Studio platform. WebSphere Studio provides content creation tools for site creation. The enablement of JSP technology for catalog display provides further evolution into a more open standards, Java environment.

The key to increasing e-commerce sales is to leverage the same proven merchandising techniques used in traditional commerce ventures. The WebSphere Commerce Suite uses an advanced rules-based framework to provide unique customer shopping experiences and to implement marketing campaigns that improve site traffic and sales. With WebSphere Commerce Suite, you can create your own marketing campaigns, or use packaged campaigns such as:

- Merchandising

Highlight related items based on cross-sell, up-sell, accessory, or substitution strategies.

- Inventory Reduction

Recommendations and discounts for overstocked items.

- Legal Policy

Sales based on country-specific legal policies.

- Time Based

Promotions based on season, special events, or holidays.

The WebSphere Commerce Suite also supports the increasingly popular auctions sales model. You can add the excitement of an auction to attract site traffic, increase sales, and to move excess inventory.

The WebSphere Commerce Suite has rich and extensible functionality. You can tailor the interface to meet your individual needs. You can integrate your e-commerce system with existing in-house systems and middleware. You can

even add on more than 100 applications for WebSphere Commerce Suite that were developed by independent software vendors.

6.3.1 Commands

Commands perform a specific business process, such as adding an item to the shopping cart, processing an order, updating a shopper's address book, or displaying a specific product page. Most commands are exported, meaning that they can be executed as HTTP requests that generate HTTP responses in the form of Web pages. Exported commands have their name included in a URL, which can be called by a button or hyperlink, such as an Add to Shopping Cart button. When a shopper clicks the button or link, the URL containing the command's name and its parameters (as name-value pairs) are sent to the system for processing. Non-exported commands are called internally by other commands.

A command represents a static piece of business process that delegates the implementation of specific pieces of business logic to tasks. This separation between commands and tasks allows different implementations of the same command to be performed depending upon merchant parameters.

For example, in a mall, a shopper can click an Add to Shopping Cart link in any store, but the implementation or manner in which the tasks for adding the selected item to the shopping cart are performed can vary from one merchant to the next.

Depending on the nature of the command, it may do any of the following:

- Process and write information directly to the database.
- Call one or more overridable functions that are assigned to process tasks to process information and write information to the database.
- Call one or more overridable functions that are assigned to view tasks to retrieve information from the database and display it as a store page.
- Call one or more overridable functions that are assigned to exception tasks to handle any exception conditions that are encountered by the command or an overridable function. The overridable functions, by default, call Net.Data macros.
- Write an HTTP response to handle any exception conditions encountered by the command or an overridable function.

6.3.2 Tasks

While commands and overridable functions are actual pieces of code, a task is not. A task is a contract between a command and an overridable function, or between two overridable functions. These rules dictate the following:

- The work that the caller expects the overridable function to perform
- The parameters that the caller passes to the overridable function
- The parameters and other results that the caller expects the overridable function to return

The following three types of tasks are defined by WebSphere Commerce Suite:

- View tasks

View tasks are used for overridable functions that display the pages that shoppers normally see during the shopping process, such as product pages, registration pages, and ordering pages.

- Process tasks

Process tasks are used for overridable functions that process information, such as calculating the total cost of an order.

- Exception tasks

Exception tasks are used for overridable functions that handle exception conditions, such as when a shopper tries to place an order before logging on as a registered shopper, or when a shopper tries to order an item that is not in stock.

6.3.3 Overridable functions

Overridable functions provide extension points with which you can customize the way your store functions. Overridable functions are written in C++ and are implemented as platform-specific CGIs for the WebSphere Commerce Suite runtime platform, such as Windows NT, AIX, Solaris, OS/400, and OS/390.

To create an overridable function, we recommend the following steps:

1. Ensure that you follow the required standards. Refer to the Commerce Suite online help system for a description of standards that should be followed when writing overridable functions.
2. Ensure that you understand tasks, overridable functions, and commands, as well as how they work together in the Commerce Suite.

3. Ensure that you understand the basic classes that are needed to write overridable functions. The Commerce Suite online help system provides a listing of these basic classes as well as their descriptions.
4. Ensure that you are authorized to replace the overridable function that is currently assigned to its corresponding task with the new function. A site administrator has the authority to create any overridable functions (for a mall as well as stores) and sets the scope of tasks. If a task is set to store scope, then a store administrator can create an overridable function to implement the task. If you are trying to create an overridable function, but do not have authority, see your site administrator to get the proper access.
5. Plan the overridable function. Begin with understanding the business logic that you are trying to implement with the new overridable function, then determine the process task that implements the required business logic. This process task has a default overridable function (to be replaced by your new overridable function) that you should examine to understand how it works, including its input and output parameters. Your planning should also cover how to handle exception conditions and ensuring that you use parameters that are passed into the overridable function and that you return all variables that the system expects to receive.
6. Write the overridable function. Two templates for overridable functions are provided. The first is a template for the CPP file (the source file, written in C++). In the CPP template, you replace instances of My Overridable Function with the name of your new overridable function and insert your customized code. Use the appropriate compiler for your operating system platform to compile your code. The second template is for the make file for the new overridable function. Refer to the WebSphere Commerce Suite online help for more information, such as the location of the template files, where to store the new files, and the correct version of the compiler to be used.
7. Register the new function in the database. Make a record of the new overridable function in the OFS database table.
8. If you set an exception task to handle exceptions that are encountered by the overridable function, change the corresponding error macro that will display the error page.
9. Activate the new function by assigning it to its task. In step 5 you determined the task that implements the required business logic, and therefore; should call the new overridable function. In this step, use the Commerce Suite Administrator to assign the new overridable function to that task. Once the assignment has been made, you must restart the Commerce Suite server for the change to take effect.

10. Depending on the change that you are making, you may need to do the following when you implement the new overridable function:
 - a. Extend the Commerce Suite database. If the new overridable function requires the use of custom data in the database, you must customize the database. This can be done by either using columns that have been reserved for merchant use, or adding new database tables. Refer to the Commerce Suite online help system for more information.
 - b. If you modified one or more error macros to handle exception conditions, activate the error macros. Use the Commerce Suite Administrator to activate the error macros.
11. Test the overridable function.

6.3.4 Database

The WebSphere Commerce Suite is implemented using a relational database. The database contains all the information that is used by the WebSphere Commerce Suite system, including information about individual shoppers, items, and prices, etc.

The WebSphere Commerce Suite database consists of a collection of tables. Every time you save a task from the forms in the WebSphere Commerce Suite Administrator, such as adding a new product or entering information in the forms, you are storing the information in one or more tables in the database. The tables are automatically updated with the new data. The database contains customizable merchant-reserved columns. Merchants can also customize the database by adding new tables.

Database features include:

- An index mechanism
- Database views
- Trigger and delete cascade mechanisms to preserve the referential integrity of the relationships between tables in the database.
- ODBC support for DB2
- ODBC support for Oracle

6.3.5 CGI implementation

WebSphere Commerce Suite supports C++ and Net.Data CGI from the legacy Net.Commerce programming model.

6.3.5.1 C++ CGI

CGI scripts can be written in interpretive languages such as Perl, TCL, REXX and UNIX shell scripts. Alternatively, CGI can be written in C++, as is the case in WebSphere Commerce Suite. When the CGI request arrives at the Web server, it forks off a new process to run the script and sets up the environment variables for this process. Included in the variables are any form input as well as information about the requestor such as the browser or its level. Any response from the CGI script is processed by the Web server according to its content. Typically the response will be an HTML stream that will then be sent back to the client and appear in the client's browser window.

The sequence of events for a CGI script are as follows:

1. A client at a browser clicks a link that contains the /bin-cgi/ phrase (positioned immediately after the host name portion of the URL).
2. The Web server intercepts the request and looks in its cgi-bin directory for the script named in the URL portion following the /cgi-bin/ keyword.
3. A process is forked off to run the interpreter that will process the script.
4. The process is passed a set of environment variables that include both system information and any parameters that were attached to the name of the cgi-bin script command name.
5. The script is interpreted (runs) and in the process of executing, usually will generate an HTML stream mixed with data to return to the client. The HTML stream is constructed on the fly per the design and function of the script.
6. The process terminates and resources are freed.
7. The client sees the resulting HTML stream as yet another Web page or item on a Web page. Page counters are a typical small CGI script item.

6.3.5.2 Net.Data

The WebSphere Commerce Suite server displays store pages dynamically by executing macros that retrieve data from the Commerce Suite database.

It communicates with the Web server using TCP/IP sockets. It also controls shopper registration, and ensures that SSL is enabled when required to protect any confidential information. Figure 16 on page 96 shows the main components of the WebSphere Commerce Suite server using Net.Data display technology.

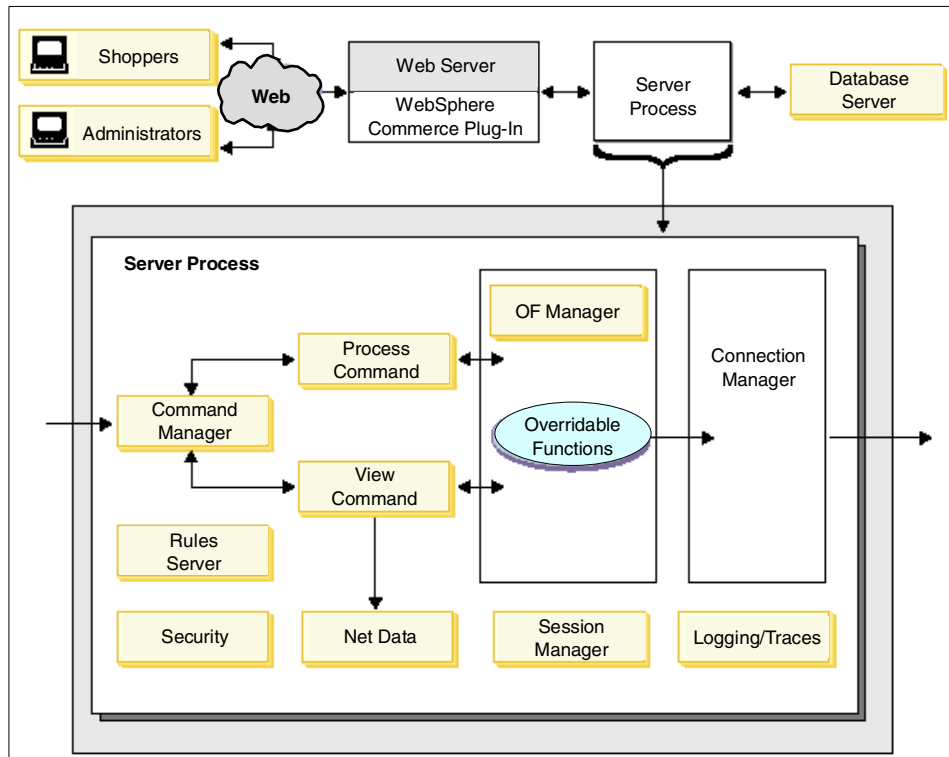


Figure 16. WebSphere Commerce Suite using Net.Data diagram

Net.Data is an interpretive programming language that is used to facilitate interaction between shoppers and the database. A Net.Data macro is a file that retrieves data from the database and displays it as a formatted Web page. It contains functions that usually execute SQL queries, HTML tags (which can also contain JavaScript code), and Net.Data statements.

In the WebSphere Commerce Suite system, a Net.Data macro is a file that retrieves data from the WebSphere Commerce Suite database and displays it as a formatted Web page. Typically, it contains functions that execute SQL queries, HTML tags (which can also contain JavaScript code), and Net.Data statements. The SQL statements search for and retrieve information from the WebSphere Commerce Suite database, the HTML defines the layout of the search results, and the Net.Data statements control the flow of the output. Net.Data macros can be created in any text editor.

6.3.5.3 Rules

Personalization in WebSphere Commerce Suite uses a rule-based system. By using rules derived from the business policies of your company, WebSphere Commerce Suite can evaluate a particular shopper or their purchases, and generate a list of products to recommend to them. The rules take the simple form of <if condition, then action>. For example, simple rules are similar to the following statements:

- If the customer has an interest in gardening, then recommend a shovel.
- If the customer has a handsaw in their shopping cart, then recommend a circular saw.

The rule processor evaluates whether the condition is true or not, and if it is, then it performs a particular action. Both the conditions to evaluate and the actions to take are highly flexible; therefore, you can capture all of your business' processes and protocols.

The rule processor is a robust engine that evaluates rules that were created in an English-like scripting language, the Blaze Advisor Structured Rule Language (SRL). SRL features all of the flexibility associated with an object-oriented approach.

After completing initial development, you can alter or replace your collection of rules without shutting down the entire site.

6.3.5.4 Personalization

Personalization helps you increase sales opportunities by allowing your e-commerce site to utilize traditional sales and marketing strategies. Many e-commerce sites mass market their products by displaying static catalogs to all their customers. Using personalization, you can initiate marketing campaigns that target particular customer groups. You can use information provided by your customers or information learned about your customers to apply relationship marketing techniques, such as up-sell and cross-sell, that help you recommend specific products to likely buyers. These strategies increase both sales and customer loyalty. Focusing on a one-to-one relationship with your customers by delivering personalized content will help to develop and keep customer loyalty.

Developing your customers' loyalty is especially important on the Web because your customers can very easily leave your store to go to a competitor. Maintaining a one-to-one relationship with a loyal customer leads to long-term, satisfied customers who will make multiple purchases over time. Personalization is one aspect of implementing a successful Relationship Marketing campaign through your online store.

6.3.6 WebSphere Application Server integration

WebSphere Commerce Suite V4.1 provides support for JavaServer Pages, Java Servlets, and JavaBeans using the WebSphere Application Server.

6.3.6.1 JavaServer Pages in WebSphere Commerce Suite

The WebSphere Commerce Suite uses JSPs to build dynamic content for catalog display from the database. WebSphere Application Server provides the runtime support for JSPs. The data beans used by the JSPs retrieve product and category information from the database. Use Page Designer in WebSphere Commerce Studio to create the JSPs.

Figure 17 shows the main components of the WebSphere Commerce Suite server using JSP display technology:

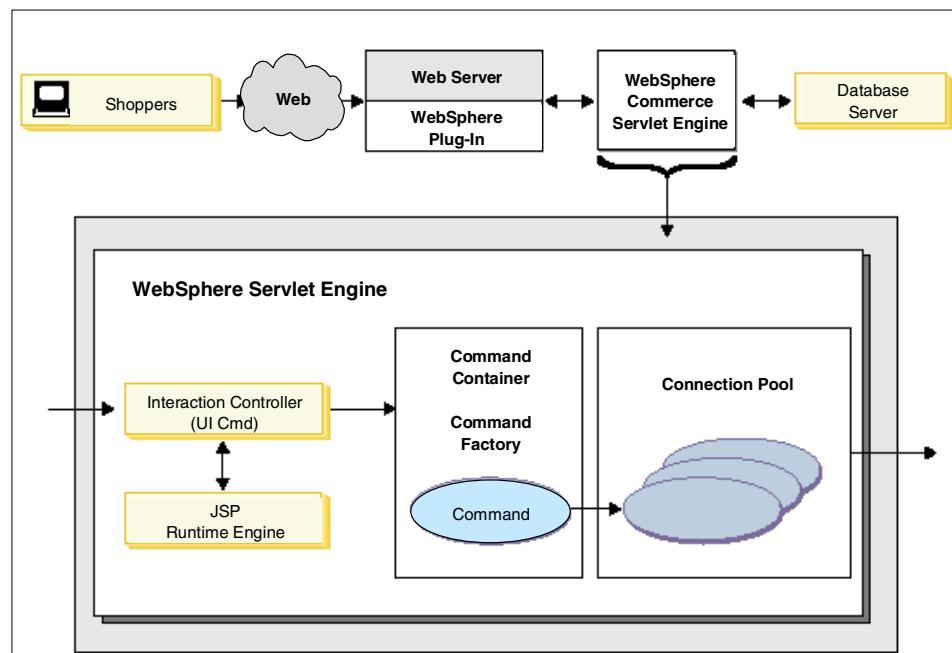


Figure 17. WebSphere Servlet Engine

JavaServer Pages (JSP) technology allows you to create HTML pages that include dynamic elements. As a result, you can use JSP to create catalog pages, which typically contain dynamic content, such as products, product prices, and attributes.

WebSphere Commerce Studio includes a set of WebSphere Commerce Suite beans that you can drag and drop anywhere on a catalog page created with

JSP technology. These beans allow you to retrieve category, merchant, and product information from the database without writing any code.

You can also add your own images, static text, tables, and other elements using Page Designer's WYSIWYG page editing function without any prior programming knowledge.

When a customer requests a JSP page, a JSP-enabled engine (in this case, the WebSphere Application Server) interprets the JSP tags and scriptlets, creates the content in the form of an HTML page, and returns it to the browser.

Using JSPs to create dynamic HTML pages allows you to separate the task of coding data retrieval from the task of creating the display format. For example a Java programmer can develop Java beans to retrieve data from the database, while a media specialist can design the look and feel of the page.

Note

You must use JSP 1.0 specifications to create catalog pages in Commerce Studio.

Note

Although WebSphere Commerce Suite supports templates created with JSPs or Net.Data, you can use only one type of technology in your store at a time. That is, if you create a product page template using JSP technology, you must create category page and product list templates with JSP technology.

6.3.6.2 Java Servlets in WebSphere Commerce Suite

In the past, server-side requests were often handled by CGI (Common Gateway Interface) programs. Servlets provide advantages over CGI programs in that they can run with less overhead, are more portable, and are easier to maintain.

The servlet engine is the function of the Web application server that manages servlets. It manages the creation and destruction of servlets, specifies which servlets should be automatically loaded into memory at startup and which should be loaded upon initial request.

In the WebSphere Commerce Suite environment, JSP files, compiled into servlets, can be used for the display of category and product pages from your catalog. In addition, you may take advantage of the extensions to the servlet API that WebSphere Application Server has provided. For example, an extension is provided that allows you to track the page that has referred visitors to your site.

6.3.6.3 JavaBeans in WebSphere Commerce Suite

WebSphere Commerce Studio includes a set of beans that provide access to information in the WebSphere Commerce Suite database tables. Creating store pages using these beans is useful to display information that may change often, such as category and product lists, or product prices. Typically you will use WebSphere Commerce Suite beans when creating pages for your catalog, such as category, product list, or product pages.

Each WebSphere Commerce Suite bean accesses information from a particular table. For example, the Category bean accesses information from the Category table. In that table, the bean accesses information from a particular column through properties. That is, the category reference number property accesses the information in the CGRFNBR (category reference number) column in the category table.

Although each WebSphere Commerce Suite bean contains distinct properties, they all share the following properties:

- Merchant reference number - the merchant reference number associated with the bean.
- User reference number - the user reference number associated with the request of the bean.
- Error message - the error message for the bean. The message is available if the request to activate this bean fails.
- Error code - the error code for the bean. The code is available if the request to activate this bean fails.

Some WebSphere Commerce Suite beans work in conjunction with others. For example, the Child Category List bean will return a list of the applicable Category beans within a particular catalog, and the Product List bean will return a list of the applicable Product beans for a particular category within a particular catalog. The Product bean includes the Product Attribute List bean and the Price bean, allowing you to display the attributes and price for that product.

Using Page Designer, you can add the WebSphere Commerce Suite bean of your choice to your store page, and select which properties you want to display from the Bean Property Selection window.

6.3.7 Connectors

e-business connectors are gateway products that enable you to access enterprise and legacy applications and data from your Web application. Connector products provide Java interfaces for accessing database, data communications, messaging, and distributed file system services.

IBM provides a significant set of e-business connectors with tool support, for CICS, Encina, IMS, MQSeries, DB2, SAP and Domino. IBM connectors are based on the Common Connector Framework (CCF). For resources on System/390, IBM is delivering native connectors based on CCF.

The command bean model allows you to code to the specific connector interface(s) of your choice while hiding the connector logic from the rest of the Web application.

6.3.7.1 JDBC and SQLJ

The business logic in a Web application will access information in a database for a database-centric scenario. JDBC is a Java API for database-independent connectivity. It provides a straightforward way to map SQL types to Java types. With JDBC you can connect to your relational databases, and create and execute dynamic SQL statements in Java.

JDBC drivers are RDBMS specific, provided by the DBMS vendor, but implement the standard set of interfaces defined in the JDBC API. Given common schemas between two databases, an application can be switched between one and the other by changing the JDBC driver name and URL. A common practice is to place the JDBC driver name and URL information in a property or configuration file.

There are four types of JDBC drivers from which you can choose, based on the characteristics of your application:

- Type 1: JDBC-ODBC bridge drivers. This type of driver, packaged with the JDK, requires an ODBC driver and was introduced to enable database access for Java developers in the absence of any other type of driver.
- Type 2: Native API Partly Java drivers. This type of driver uses the client API of the DBMS and requires the binaries for the database client software. This type of driver offers performance advantages but introduces native calls from the JVM.

- Type 3: Net-protocol All Java drivers. A generic network protocol is used with this type of driver. Portability is a major advantage of this type of driver, but it has the limitation that it requires intermediate middleware to convert the Net-protocol to the DBMS protocol.
- Type 4: Native-protocol All Java drivers. This type of driver is portable and uses the protocol of the DBMS. Type 3 and 4 drivers are well suited for applets that access a database server on an intranet, as they only require Java code to be downloaded.

An important technique used to enhance the scalability of Web applications is connection pooling, which may be provided by the application server. When application logic in a user session needs access to a database resource, rather than establishing and later dropping a new database connection, the code requests a connection from an established pool, returning it to the pool when no longer required.

SQLJ provides a simplified syntax for JDBC that allows you to write SQL-like statements directly in your Java source code. The SQLJ preprocessor generates static SQL providing better performance than dynamic SQL. SQLJ will also generate iterator Java classes. These iterators allow you to navigate query results using a very simple “get next” protocol.

As your design takes shape, based on its desired performance and sophistication you may see the need to investigate SQLJ or enterprise Java beans. The most recent level of the JDBC specification is 2.0, but many JDBC drivers you use will still implement 1.0.

6.3.7.2 MQSeries Adapter

The WebSphere Commerce Suite MQSeries Adapter supports a set of defined outbound and inbound messages that help integrate WebSphere Commerce Suite business processing with back-end system business processing. The MQSeries adapter offers MQSeries messaging support for messages in XML format. When you use these messages, they invoke specific behaviors within the WebSphere Commerce Suite server. The format of the XML messages consists of a set of XML elements defined within specific DTD files for each XML message.

6.3.8 Mass import XML

The Mass Import Utility has been improved to support XML and cover schema changes introduced in WebSphere Commerce Suite V4.1. Figure 18 depicts how the mass import utility works.

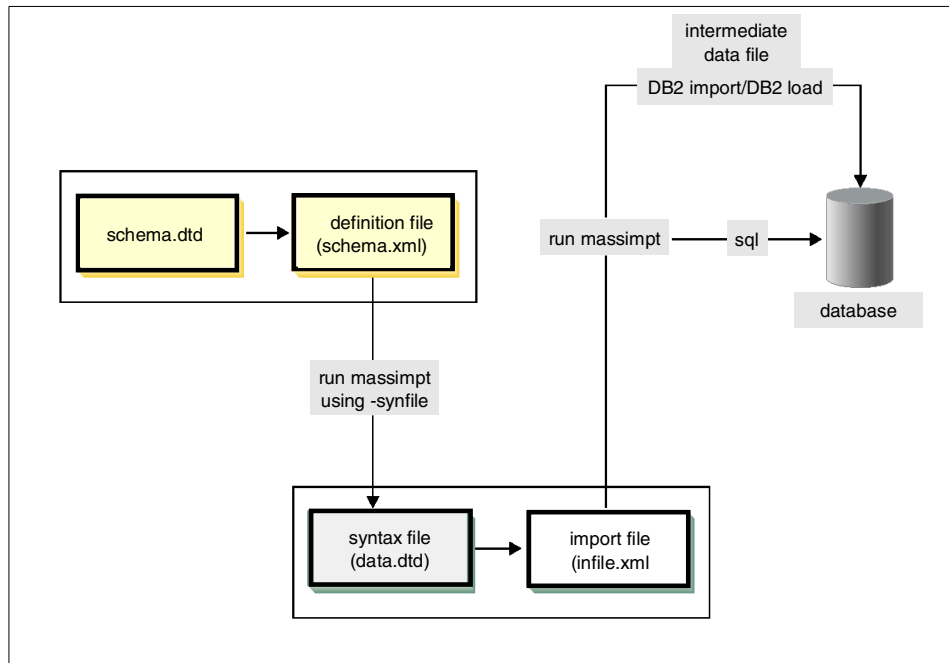


Figure 18. Mass import using XML import files

6.3.9 Additional enterprise Java APIs

In developing a server-side application, you may also need to be familiar with the following enterprise Java class libraries:

6.3.9.1 Java Naming and Directory Interface (JNDI)

This package provides a common API to a directory service. Service provider implementations include those for LDAP directories, RMI and CORBA object registries. Sample uses of JNDI include:

- Accessing a user profile from an LDAP directory
- Locating and accessing an EJB Home

6.3.9.2 Remote Method Invocation (RMI)

RMI and RMI over IIOP are part of the EJB specification as the access method for clients accessing EJB services. RMI can also be used to implement limited function Java servers.

6.3.9.3 Java Message Service (JMS)

The JMS API enables a Java programmer to access message-oriented middleware such as MQSeries. Such messaging middleware is a popular choice for accessing existing enterprise systems.

6.3.9.4 Java Transaction API (JTA)

This Java API is for working with transaction services, based on the XA standard. With the availability of EJB servers, you are less likely to use this API directly.

6.4 Where to find more information

IBM Publications:

- *Servlet and JSP Programming with IBM WebSphere Studio and VisualAge for Java*
- For information on the IBM Application Framework for e-business:
<http://www.ibm.com/software/ebusiness/>
- For information on the *IBM Application Framework for e-business Architecture Overview: Understanding Technology Choices* white paper:
<http://www.ibm.com/software/ebusiness/buildapps/understand.html/>
- For information on IBM XML technology: <http://www.ibm.com/developer/xml/>

Other Publications:

- Flanagan, David, *JavaScript: The Definitive Guide*, Third Edition, O'Reilly & Associates, Inc., 1998
- Maruyama, Hiroshi, Kent Tamura and Naohiko Uramoto, *XML and Java: Developing Web Applications*, Addison-Wesley 1999
- Flanagan, David, Jim Farley, William Crawford and Kris Magnusson, *Java Enterprise in a Nutshell*, O'Reilly & Associates, Inc., 1999
- For information on the Java technology: <http://www.javasoft.com/products/>
- For information on Sun XML technology: <http://www.sun.com/xml/>.

Chapter 7. Application design guidelines

e-business application design presents some unique challenges when compared to traditional application design and development. The majority of these challenges are related to the fact that e-business applications are primarily used by an undefined set of internal and external users such as employees, customers, and business partners. Web applications must be developed to meet the varied needs of these users.

This chapter is organized into the following sections:

- General Web application design guidelines
- Application elements
- Application structure
- Working logic
- Controller
- Page construction
- User tracking and sessions management
- Security
- e-commerce models
- Where to find more information

7.1 General Web application design guidelines

When designing a Web application, several key points need to be considered:

- Web browser client

The application design is determined by the Web browser client application. The e-commerce solution relies on the thin client concept.

- Stateless operation

The conversation between the client and server on the Web is stateless by default. There is no information about the user's operations.

- Know the users

The Web provides anonymity for the users and user information is hidden from the server applications.

- Who are the users on my Web site?
- Where are they on the Web site?

- Security planning
Select the appropriate level of security from login/password security to a sophisticated certificate authority solution.
- Timeless
Provide short-wait, real-time responses. The workflow needs to be aligned to the Web application behavior.
- User interaction
Everything has to be reachable using Web pages. The site map needs to be simple and traceable. Take into consideration that there are many browser implementations and some browsers are not fully compatible.
- Look and feel
Serve the customer with a Web site that provides good look and feel.
- Load
Optimize the workload, served pages, and database queries on the server, client, and network bandwidth.
- Push everything to the server side
The client applications should be used for interaction. Browser extensions such as plug-ins and applets are used to create good look-and-feel user interfaces. Some application security and communication with the server is necessary, but should be held to a minimum.

7.2 Application elements

The application uses many servers and working modules. Some of the servers and modules are essential, some are rewriteable, and others are customizable.

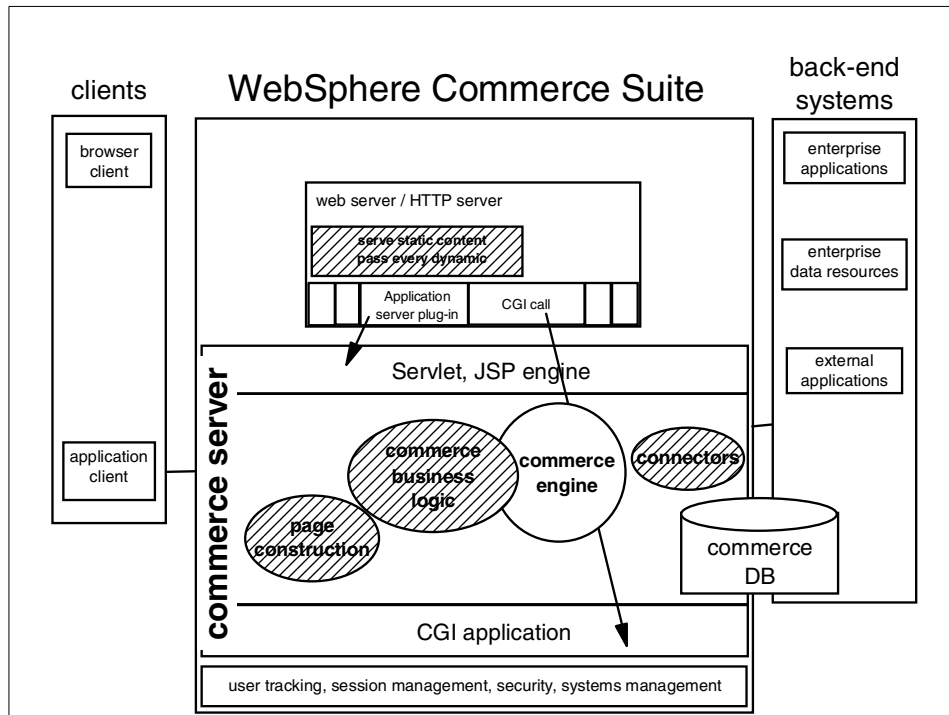


Figure 19. Key elements of User-to-Online Buying Web applications

The dashed areas in Figure 19 denote customizable elements in the system. Developers can create custom applications with these elements, which are based on the commerce engine and business logic.

The commerce application is positioned between the client-side and back-end systems. The WebSphere Commerce Suite provides a front end to the back-end system. The three pillars in Figure 19 represent the end-to-end solution for an e-commerce system. The commerce engine is responsible for all the commerce logic, strong built-in mechanisms, and many customizable elements.

With WebSphere Commerce Suite there are two different ways to approach the business and presentation logic:

1. Net.Data, C++ modules
 - This method relies on Net.Commerce legacy CGI modules.
2. JavaServer Pages, Java Servlets, and JavaBeans
 - This method is based on the WebSphere Application Server.

7.2.1 Web clients

Web clients are responsible for accepting and validating the user input, communicating the user inputs to the Web application server, and presenting the results received from the Web application server to the user. These clients can be broadly classified into application clients and Web browser clients, as discussed in 6.1, “Web clients” on page 77.

7.2.1.1 Benefits of browser clients

The WebSphere Application Server supports both Web browser clients and application client models. However, browser clients provide the following benefits compared to application clients:

- The majority of the presentation logic and all of the business logic will reside on the server. Hence it is easy to make the necessary changes to support a broad range of client devices.
- The client part of the application is lightweight and downloads quickly.
- It is easier to secure, scale and maintain presentation and business logic that resides on the server.
- Client applications that use Java applets and Java applications require a particular version of Java to be supported by all clients, thus limiting the universal accessibility of the applications.
- Client applications that use Java applets and Java applications are downloaded to a user’s local machine and can be decompiled to view the business logic. This poses security concerns. Hence critical business logic should not be directly coded in these downloadable applications.

For these reasons, the majority of Web applications being designed today are zero maintenance HTML clients. Further details on these client models of the IBM Application Framework for e-business can be found at:

<http://www.ibm.com/developer/features/framework/framework.html/>.

7.2.2 WebSphere Commerce Suite

The WebSphere Commerce Server is comprised of the following major server components:

- Web/HTTP Server
- WebSphere Commerce Server
- WebSphere Commerce Database
- WebSphere Application Server

7.2.2.1 Web/HTTP Server

The Web Server or HTTP Server is responsible for receiving and responding to HTTP requests. The HTTP server is capable of serving static HTML pages without help from other sub components of the Web application server. All dynamic page requests are passed to the application server plug-in.

- Application server plug-in

The application server plug-in provides the connection between the Web/HTTP Server and services of the WebSphere Application Server.

- CGI call

WebSphere Commerce Suite commands, tasks, and overridable functions are executed on the HTTP server using the Common Gateway Interface (CGI). Incoming request are sent to the HTTP server, which then calls the CGI application. This is a legacy technique for server-side programming provided by the Net.commerce CGI implementation.

7.2.2.2 WebSphere Commerce Server

The WebSphere Commerce server is comprised of the following components:

- CGI application

The Web/HTTP server calls the CGI application, which runs the commands, tasks, and overridable functions related to the commerce engine.

- Commerce engine

The commerce engine is comprised of all the logic, functions, tasks related to the commerce system. It has many built-in functions and procedures to control the commerce processes.

- Page construction

Page construction services provided by the WebSphere Application Server support Java-based technologies such as Java Servlets and JavaServer Pages (JSPs) for dynamic page construction.

- Commerce business logic

Business logic services provide a robust environment for processing business logic independent of the user interface client types. The WebSphere Application Server supports components based on technologies such as JavaBeans and Enterprise JavaBeans (EJBs) for programming business logic.

- Connectors

Connectors are components that support the communication between the application server and the back-end system such as databases, legacy applications and business partner applications. The WebSphere Application Server provides a number of connectors including JDBC drivers, JNDI class libraries, CICS connectors, MQSeries connectors, and IMS connectors. An example of MQSeries connectors can be found in Chapter 17, “Back-end integration using MQSeries XML messages” on page 401.

7.2.2.3 WebSphere Commerce Suite Database

The commerce database contains information related to the commerce applications, including products, customers, store information, etc. Almost all information for the commerce site is stored in this database.

WebSphere Application Server (WAS)

WAS provides the focal point of the Web application technology, including the following responsibilities:

- Java server-side programming model

To meet these requirements, WAS provides a range of dynamic page construction, business logic processing, and data access in the form of JavaServer Pages, Java Servlets, and JavaBeans.

- Session management

Session management services are necessary because inherently HTTP is a stateless protocol. In order to address the issues of a stateless protocol, a number of HTTP session management techniques have been developed.

- User tracking

The WebSphere Application Server has an API called UserProfile that makes it easy to maintain persistent information about your Web site visitors. The UserProfile API enables you to store the user's data in a database without coding any SQL. UserProfile contains the interface to access the user's personal data, such as name, postal and e-mail addresses, telephone numbers, and shopping cart information.

- Security

Security services include authentication, authorization, data integrity, privacy (encryption) and non-repudiation services. The WebSphere Application Server provides these services by supporting industry-standard protocols such as SSL, LDAP, etc.

- Systems management

Systems management services provide a robust runtime environment for the application hosted in the Web application server. These services allow load balancing, fail-over support, remote monitoring, etc. The WebSphere Application Server supports many systems management features. Further details on systems management can be found in Chapter 9, “System management and security guidelines” on page 187.

7.2.3 Back-end systems

Back-end systems include enterprise data sources, existing or new enterprise applications. For example, ERPs, inventory management, financial systems, and business partner systems.

7.3 Application structure

A Web application can be viewed as a set of interactions between the Web browser clients and the Web application server. The interaction begins with an initial request by the browser for the welcome page of the application. All subsequent interactions are initiated by the user by clicking on a user interface control, such as a button or a link. This causes a request to be sent to the Web application server. The Web application server processes the request and dynamically generates a result page and sends it back to the client. A closer examination of these interactions reveals a common set of processing requirements that need to be considered on the server-side. These interactions can be mapped to the classic Model-View-Controller design pattern as shown below.

- Model

Model represents the application object that implements the application data and business logic.

- View

The view is responsible for formatting the application results and dynamic page construction.

- Controller

The controller is responsible for receiving the client request, invoking the appropriate business logic, and selecting the appropriate view to be presented to the user, based on the results.

For more detailed information about Model-View-Controller design, refer to the *Patterns for e-business: User-to-Business Patterns for Topology 1 and 2 using WebSphere Advanced Edition*, SG24-5864, in section 7.3.1.

7.3.0.1 Commerce application structure

In simple terms the commerce application could be defined as follows:

- The commerce server is responsible for running commerce processes within the application server (for example, order processing).
- The Web server serves pages to the client (for example, catalog pages).
- The database stores information about the commerce application (for example, products).
- The back-end system is responsible for the business processes beyond the commerce processes (for example, inventory management).

The commerce server connects to the database and the back-end system using the application server. The information provided by the two systems is maintained and processed by the commerce server and sent to the client Web browser through the Web server.

7.4 Working logic

There are two approaches to build the interaction from business components to HTML pages in WebSphere Commerce Studio:

- Net.Data macros
- JavaServer Pages

The next two sections examine the presentation, business logic, and controllers using both Net.Data macros and JavaServer Pages.

7.4.1 Net.Data, C++ modules

The legacy commerce application design is CGI based, using Net.Data and C++ modules. Figure 20 on page 113 shows how the user interacts with the WebSphere Commerce Web site using the Net.Data and C++ CGI approach.

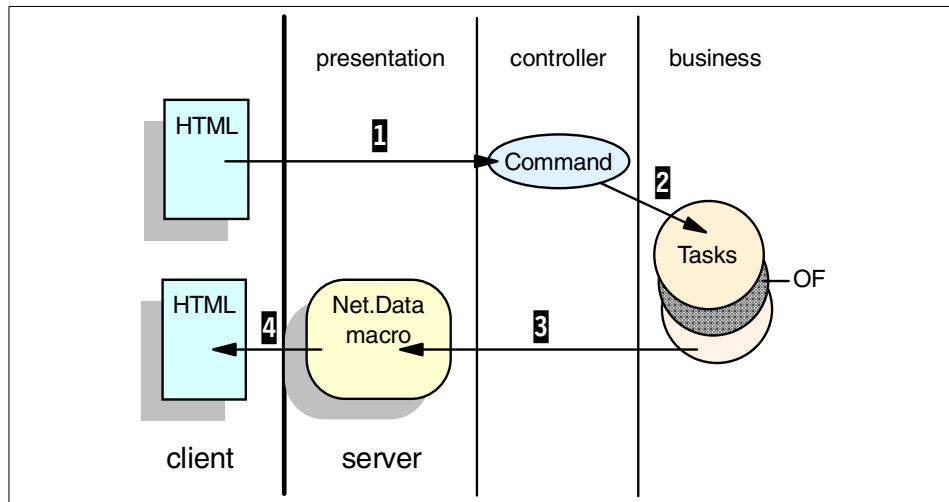


Figure 20. Net.Data, C++ modules approach

1. From the HTML document a URL or a FORM action points to a command on the server which belongs to the WebSphere Commerce Server.
2. The command behaves as a controller and starts the processing by calling the tasks. The tasks are processed one after another, and each task represents an element of the business logic. Some of the tasks are customizable, and can be replaced with a new function, called Overridable Functions (OFs).
3. The last task (view task) executes the appropriate Net.Data macro.
4. The Net.Data presents the content, which is interpreted by the WebSphere Commerce Server and is sent back to the user as an HTML page.

7.4.2 JavaServer Pages, Java Servlets and JavaBeans

The Java-based WebSphere Application Server provides another technology for the same business logic-controller-presentation logic triad. Figure 21 on page 114 shows how the user interacts with the WebSphere Commerce Web site using the JavaServer Pages, Java Servlets, and JavaBeans.

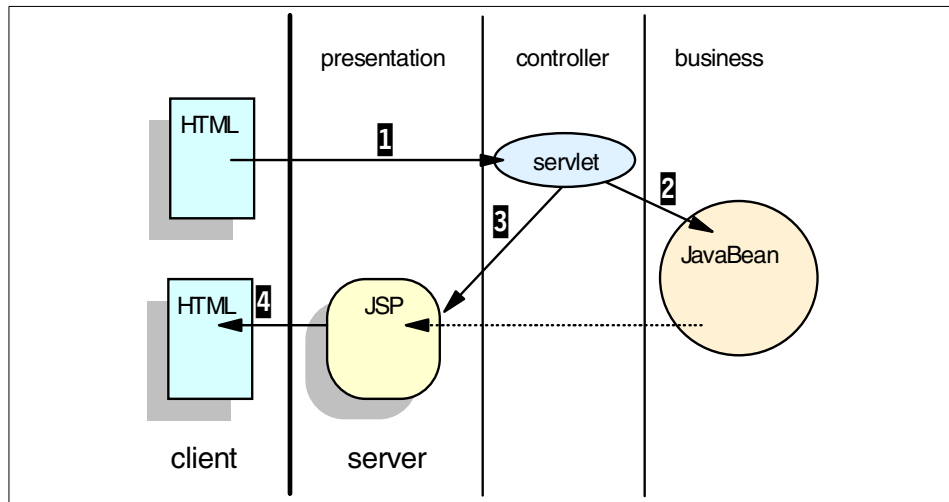


Figure 21. Java based approach

1. From the HTML document a URL, or a FORM action points to a servlet on the server which belongs to the WebSphere Commerce Server. The servlet call in this context is equivalent to the command call.
2. The servlet is responsible for the control of the request, by instantiating the JavaBeans, passing the parameters as a property, and executing the methods. The JavaBeans represent the business logic elements.
3. Next, the servlet calls the JSPs, which are interpreted by the WebSphere Application Server. The JSP instantiated JavaBeans allows access to the business logic information.
4. The JSP sends the generated HTML to the Web browser client.

7.4.3 Business logic

The business logic code is ultimately responsible for satisfying the client requests. As a result, business logic must address a wide range of potential requirements, which include:

- Ensuring transactional integrity of application components
- Maintaining and quickly accessing application data
- Supporting the coordination of business workflow processes
- Integrating new application components with existing applications

7.4.3.1 WebSphere Application Server - business logic

To address these requirements, the WebSphere Application Server supports business logic written in and using the full facilities of the Java runtime, including support for:

- Java Servlets
- JavaBeans
- Enterprise JavaBeans (EJB)
- JDBC
- CORBA
- LDAP
- Connectors to MQSeries, CICS, IMS and other enterprise services

A detailed discussion on how business logic should be developed is beyond the scope of this book. It is valuable to consider the interface between the Web parts of the interaction (interaction controllers and the display pages) and the business logic. Such a separation of business logic from the Web-specific interaction controller and display page logic isolates the business logic from the details of Web programming. This increases the reusability of the business logic in both Web and non-Web applications.

7.4.3.2 WebSphere Commerce - business logic

Business logic within the WebSphere Commerce Suite is implemented by using commands, tasks, and overridable functions.

Commands

Think of commands as something that you would write once and never touch again. Commands encapsulate a particular business process.

Tasks

Tasks exist to enable potential variations in the business process.

Overridable functions

Overridable functions (OFs) implement variations of the tasks.

If you find yourself modifying a given command over and over again to fit various situations, then you probably have not designed it properly. The command should be generic to allow it to be varied by defining different tasks.

Commands work like a method in a base class that implement a general algorithm, which calls a pure virtual function that will get implemented at a later time. This is a pattern typical of a framework environment.

In summary, commands and overridable functions are the components that make up the customizable runtime of a WebSphere Commerce Server. They represent the business process and logic involved in running an e-commerce Web sites. Commands invoke OFs indirectly through Tasks that define the interfaces. OFs are implemented separately from commands, and are “plugged in” by a WebSphere Commerce Suite administrator. Commands and OFs share the same programming model, and are managed similarly at the system level. For more information refer to the *IBM WebSphere Commerce Suite Pro Edition, Commands, Task, Overridable Functions and Database Tables V4.1* product guide.

JavaBeans

The JavaBeans architecture provides reusable software components that can be manipulated visually by builder tools. The code written to this architecture, supplies specialized Java classes that support the following features:

- Properties
- Methods
- Events
- Introspection
- Persistence

Properties are attributes that have set and get methods (Java method) that can be called from other components. Events define a framework for one component to notify the other when something noteworthy happens. The JavaBeans specification defines a set of conventions for defining properties, methods, and events. Using this convention, builder tools can analyze the bean to allow for visual manipulation. In addition, beans should implement the persistence mechanism allowing the customized JavaBeans state to be stored and retrieved when necessary. JavaBeans can be either visual or non-visual; however, they should all allow manipulation by visual builder tools. This is usually done by using Java introspection techniques.

7.4.4 Presentation logic

The presentation logic is responsible for generating the “View” component in the form of an HTML page that will be returned to the Web browser client. The WebSphere Commerce Suite allows display pages to be implemented as either JavaServer Pages or Net.Data macros. JSPs and Net.Data macros allow template pages to be developed directly in HTML, with scripting logic inserted for dynamic elements on the page.

In many cases, the interaction controller will pass the dynamic data to the display page for formatting. In other cases, the display page will invoke business logic directly to obtain dynamic data. It makes sense to have the interaction controller pass the data when:

- It has already obtained the data
- The data is an essential component of the contract between the interaction controller and the display page

In other cases, the data needed for display is not an essential part of the interaction and can be obtained independently by inserting calls to business logic directly in the display page.

The WebSphere Commerce Suite allows you to prepare pages with Net.Data macros, and some pages with JavaServer Pages. The design considerations about these two technologies are as follows:

- Net.Data macros

Many people still use Net.Data to implement business logic due to this being the legacy technology. Although Net.Data could be considered adequate to develop display pages (that is, only issue SELECT SQL statements and format the results into HTML), they do not match the security, power, simplicity and convenience of commands to implement business logic. If you use Net.Data to implement business logic, you forego access control, access to a real programming language, access to all the features available to commands such as the e.commerce Data objects that isolate you from SQL, access to INI variables, better performance, and the possibility of a richer error-handling capability.

We recommend that Net.Data macros only be used for formatting information in HTML. Fundamentally, the issue is not to mix presentation logic with business logic.

- JavaServer Pages (JSP)

JSPs provide a simple yet powerful mechanism for inserting dynamic content into Web pages. The JSP specification achieves this by defining a number of HTML-like tags that allow developers to insert server-side Java logic directly into HTML or XML pages that are sent to the Web browser clients. Hence, JSPs automatically get access to certain implicit objects without having to declare them.

Note

It is important to note that JSP technology is an extension of the Java Servlet API. In fact, application servers compile JSPs into HttpServlets before executing them. Essentially, the JSPs represent the service() method of the automatically generated HttpServlet.

When communicating with servlets, in order to get information from the business logic, you should use JavaBeans as message exchangers.

It is also possible to write inline Java code into your JSP. The Java code will be compiled directly into the servlet. In fact you can write highly sophisticated code in a JSP, but it is not recommended. Try to use your JSPs as simple data formatters, just like the Net.Data macros.

Note

In other books, you may find different a view of the presentation logic. In their opinion the presentation logic is only the HTML documents that are sent to the Web browser. They describe JSPs and Net.Data macros among the controllers. We want to distinguish between developing the business logic, and developing the presentation logic. In this context the JSPs and Net.Data macros belong to the presentation logic.

7.4.5 JavaServer Pages or Net.Data macros

Which technology is better for implementing the presentation logic? We have provided some guidelines to consider:

- Usage of JSPs in WebSphere Commerce Suite V4.1 are available only for category display, product list display, and product display pages, while Net.Data macros are available for all pages.
- At this time JSPs might be slower than Net.Data macros, but there is work being done to enable JSP caching with the WebSphere Commerce Suite. This functionality may be provided in a service pack or the next release. Once the page is cached, upon first usage, any further request will be served faster than Net.Data.

For different kinds of caching and performance considerations refer to Chapter 5, “Performance guidelines” on page 53.

- Net.Data macros are based on an interpretive scripting language and executed as CGI applications. Net.Data generated pages are cached by WebSphere Commerce Suite.
- JSPs are Java-based pages, parsed by the server and compiled into servlets. This approach accesses the latest technology on server-side programming. Future releases of WebSphere Commerce Suite will fully implement servlets and JSPs for all pages supported by Net.Data.
- A designer tool for Net.Data macros does not ship with WebSphere Commerce Studio.
- WebSphere Commerce Studio provides a WYSIWYG designer for JSP generation.
- With JSP pages you can use pure Java code. With this capability you can access the Java-based application server and do further development.
- Net.Data does not allow you to access the Java-based application server. You have to develop your own C++ modules.

7.5 Controller

Java Servlets provide the connection between the JavaServer Pages (presentation logic) and the commerce JavaBeans (business logic).

Servlets are protocol and platform independent server-side Java components. They implement a simple request and response framework for communication between the client and the server. The Java Servlet API is a set of Java classes that define a standard interface between Web clients and the Web application server. The API is composed of the following two packages:

- `javax.servlet`
- `javax.servlet.http`

The `javax.servlet` package implements the generic protocol-independent servlets. The `javax.servlet.http` package extends this generic functionality to include specific support for the HTTP protocol.

7.6 Page construction

There are three different methods to provide HTML pages from the server to the Web browser client in WebSphere Commerce Suite.

1. Static content

The static content is provided by HTML documents. These documents can embed graphics, audio, video, applet, etc. There is no dynamic element on these pages.

2. Dynamic content

There are many ways to create dynamic content on the server, depending on the technology used for generating the final HTML document. One way is to directly generate the HTML page. The HTML code is wired into the programming code. This is not a flexible solution, so it is difficult to change the content once you have programmed the module.

3. Templates + dynamic data

By using a scripting language with templates to provide the skeleton of the document, the controlling functions can present dynamic data on the HTML page from different data sources.

7.7 User tracking and session management

For the purposes of this discussion, a session is defined as a series of requests originating from the same user, and the same Web browser. HTTP is a stateless protocol by design. Over the years a number of techniques have been developed to maintain the application state across multiple HTTP requests from the same user. Cookies and URL rewriting are the most commonly used techniques. The WebSphere Application Server implements the HttpSession API that hides the complexity of these techniques from the Web application programmer.

The WebSphere Application Server provides various configuration options for session management. These configuration options can influence the application behavior, performance, and failover capabilities. We urge you to consider these options early in the design phase.

Further details on user profile issues can be found at:

http://www.ibm.com/software/webservers/appserv/doc/v30/se/web/doc/begin_here/index.html/

7.7.1 Cookies and URL rewriting

The WebSphere Application Server can be configured to use cookies, URL rewriting, or both to maintain sessions across multiple HTTP requests. With both mechanisms WebSphere creates a session ID to identify a session uniquely. With cookies, the session ID is sent back to the browser as a field inside the cookie. With URL rewriting the session ID is appended to the URL and sent back to the browser. During subsequent requests the browser sends the session ID back to the server as part of the cookie or the URL. The server

is responsible for retrieving this session ID from the HTTP request and using it to obtain the proper HttpSession for the Web browser client.

Early in the high-level design phase it is important to decide whether your application should be developed to support cookies, URL rewriting, or both. Such a decision should be made based on the demographics of the end users of the system. Some users configure their browsers not to accept cookies. If you suspect this may be the case, consider supporting the URL rewriting option. For the majority of applications we recommend using only cookies.

The decision to support URL rewriting impacts the code development significantly. This decision also has certain performance implications, since all display pages, including static pages, have to be converted to JSPs or servlets. This adds unnecessary runtime processing. This also means that all static pages that could have been hosted by a Web server now need to be moved to the Web application server node since all pages have to be converted to JSPs or servlets.

7.7.2 Session persistence and clustering

Web application availability and performance can be increased by adding duplicate Web application server nodes using a *load balancer* to distribute Web requests across multiple Web application servers. Under such a scenario, if one Web application server fails, the load balancer would recognize this event and forward all the subsequent requests to the remaining Web application servers.

The above scenario can be extended to provide *failover* support. This is achieved by enabling WebSphere Application Server session persistence and session clustering.

When session persistence is enabled the WebSphere Application Server stores all session data in a JDBC-compliant relational database. The session persistence is automatically managed by the WebSphere Application Server. Application programmers need not write any special code for this session persistence to occur. However all objects that are being inserted into the session pool must implement the *serializable* interface.

Session clustering is a mechanism where more than one instance of the application servers share a common session pool. A cluster is the binding of two or more application servers that reside on separate nodes. This allows servlets to execute on any one of these nodes and have access to session data that was created by another node. The WebSphere Application Server

exploits its session persistence feature to implement session clustering. In order to enable session clustering, session persistence must be turned on.

In designing systems that exploit session persistence and clustering features, we provide the following guidelines:

- Session persistence is implemented using a generalized persistence mechanism in order to allow for various types of information to persist in the session pool. Storing large amounts of data in a generalized session pool could result in performance degradation. The session pool should only be used to store data that is essential for subsequent transactions.
- All objects that must be propagated across the cluster along with the session must be serializable. We recommend implementing the serializable interface for all objects that you anticipate being stored in the session pool. This allows for an easy transition of your applications to a clustered environment.

For information on session management, refer to:

http://www.ibm.com/software/webservers/appserv/doc/v30/se/web/doc/begin_here/index.html/.

7.8 Response time

Web applications are client-server applications, even if architectural they are n-tier systems. In the context of transactions between the client and the server, response time is a major factor. Response time is the time between the request demand and the response message on the Web client.

7.8.0.1 Processing requests

Every time users access a URL, the server has to respond within an acceptable period of time, for static pages or dynamically generated pages.

The commerce workflow has to be planned to be consistent with the Web application possibilities. If the user requests something that takes a long time to deliver information, you need to provide the user with the ability to view the status of the request. At a minimum, you need to acknowledge the transaction. For example, a customer orders from the catalog. The site acknowledges that the order is sent, but nothing else. You can send automated mail to the customer about the order processing, or prepare a personalized page where the customer can follow the order process.

7.8.0.2 Quick response time

Response time is a performance topic that needs to be considered in the application design. From the application design side you only have to take

care about optimizing your processes. For more information refer to Chapter 5, “Performance guidelines” on page 53.

Note

Internet users expect fast response times and do not tolerate waiting. Poor response time could result in customers visiting your competitor’s Web site.

7.9 Security

An important aspect of a successful e-commerce site is security. Your customers will be concerned with the security of their personal information as it is transmitted across the Internet and used throughout your order processing environment. In addition, you should be concerned with securing your information assets and systems.

The WebSphere Commerce Suite provides features to help you implement your security strategy. The security topics discussed in this section include:

- Authentication
- User registry
- Access control
- Integrity
- Cross-referencing
- Application server

7.9.1 Authentication

In order to conduct business online, two (or more) parties typically interact. The concept of authentication is the ability for these parties to trust the identity of the other (in other words, each party is who they say they are). The WebSphere Commerce Suite provides the following two modes of authentication:

- Basic authentication
- X.509 authentication

If you select to use the basic mode of authentication (the default mode), users have a logon ID and password registered in the WebSphere Commerce Suite server user registry.

The WebSphere Commerce Suite supports client certificate logon as a security mechanism, protecting both the site and the customer. The X.509

certificate supplements basic authentication for customers entering a site. A customer holding this certificate can access a secured WebSphere Commerce Server site, which has been enabled for client certificate authentication.

Before you can begin using X.509 certificates, you must arrange for a trust relationship with external certificate authorities to handle electronic authentication of X.509 certificates. Certificates are provided by certificate authorities (companies) found on the Web.

7.9.2 User Registry

Many commerce sites require users to register before they are allowed to fully interact with the features of the Web site. The WebSphere Commerce Suite allows you to choose between using LDAP (Lightweight Directory Access Protocol) or using the WebSphere Commerce Suite database and commands for user registration. LDAP can be used for both authentication and user profiles. For more information about the WebSphere Commerce Suite database and command for user registration, refer to the WebSphere Commerce Suite online help.

Lightweight Directory Access Protocol - LDAP

LDAP is a client-server protocol for accessing a directory service. It was initially used as a front-end to X.509, but can also be used with stand-alone and other types of directory servers. LDAP can be used as a centralized information repository to support information sharing among various clients.

LDAP provides a standard way to authenticate users and manage information. This allows you to create a solution in which a user can register. LDAP is implemented within the WebSphere Commerce Suite by using the SecureWay Directory product. For more information, refer to Chapter 14, "SecureWay Directory (LDAP) - AIX platform" on page 289.

7.9.3 Access control

It is important to set up access controls on the WebSphere Commerce Suite commands. First, determine whether the command is meant for all shoppers, a certain group of shoppers, or administrators. For example,

- All shoppers - access control is not required
- Specific shopper groups or administrators - access control is required

All commands issued through a URL can contain the optional parameter `merchant_m`. If this parameter is specified, its value is the merchant reference

number. The behavior for this optional parameter provides one of the following characteristics:

1. WebSphere Commerce Suite separates commands into two different categories:
 - Commands that require resources specific to a merchant (for example, the `CategoryDisplay` command)
 - Commands that do not depend on any merchant resources (for example, the `AddressForm` command).

The command security form within WebSphere Commerce Suite allows administrators to specify SSL enablement for a command and user authentication. Due to the internal distinction of WebSphere Commerce Suite commands, a command such as `CategoryDisplay` would produce the correct merchant-specific command security behavior whereas a command such as `AddressForm` would give the mall level command security behavior. To ensure that the merchant-specific command security behavior is always applied, include the `merchant_rm` parameter and value.

2. If a command such as `ProductDisplay` has its command security set such that authentication is required, then if the command is executed via non-authenticated mode the user is directed to the merchant-specific `LOGON_ERR` exception task page or the mall level `LOGON_ERR` exception task page. To direct users to the merchant-specific `LOGON_ERR` page, include the `merchant_rm` parameter and value. You can set up the merchant-specific `LOGON_ERR` task page by using the Task Management form within WebSphere Commerce Suite Administrator.
3. The Web is an open environment, whether or not your site is on an intranet. Users can observe URLs on their browsers. They can modify the URLs to try to confuse the system by changing the values for some parameters, re-issue the same command multiple times, or issue a given command in a different context. Finally, always remember that only trivial systems can be made absolutely safe, and although we have documented some issues, there are potentially many others that we are unaware of at this time. By having a solution that is well designed; however, you could be in a position where you could respond quickly to a security exposure, and implement a fix rapidly. In summary, here are some known issues that you can guard against:
 - Your command must have integrity
 - Cross-reference parameters
 - Proper access control defined
 - Guard against common attacks

7.9.4 Integrity

Integrity means that a command needs to be executable with the system in any state, although most of the time a command will be designed to work in concert with other commands. Commands can potentially be accessed outside of the navigation that you intend by a user entering a URL to a command directly. Always make sure that your environment is in proper state, or that you can bring the system back into a proper state, before continuing. For instance, the `OrderDisplay` command will always recompute and lock the order completely before allowing a view to display it. You could execute this command at any time during the shopping process, and the result would always be correct. The `OrderProcess` command can only be called if the `Order` has been properly locked first, and if it is called in any other condition, an error is returned.

You also have to be prepared for the repeat scenario in which the same command is called more than once in a row. It is possible for a user to use the Back button, and re-issue the same command a second, third or fourth time. Be sure the semantics of such an action are well defined. For instance, calling `OrderProcess` a second time causes an error, whereas calling the `OrderItemAdd` command a second time adds the product to the order again.

7.9.5 Cross-referencing

Cross-referencing means that all the data you receive should be checked against each other. For instance, a command could receive a product number, and a merchant number, and assume that the product mentioned belongs to the merchant passed. Do not assume; make sure. When you look up the product from the database, simply use the merchant ID to further constrain the query. If the two parameters do not agree, then you have a system error and you can return a false message. In this case either the calling page was not coded properly, or someone is playing around with the site, which means that it is OK to return the system error page.

For another example, consider a ticketing system where particular events are modeled using categories. When a user requests tickets, you might have a situation where your back-end system expects the tickets to all belong to the same event (that is, you cannot order tickets for multiple events at the same time). In this case, you have to cross reference the products you receive to make sure they belong to the same event. Never think that the set of pages put together to access your site is the only way that a user can call your commands.

7.9.6 WebSphere Application Server

The WebSphere Application Server provides a robust security model for securing Web application resources. Key components of the security architecture include:

- Security plug-in
- Security collaborator
- Security application
- Security server

The security model supports various authentication and authorization techniques. For more information on this topic, refer to Chapter 9.3, “Security guidelines” on page 207. In addition, you can find security information about the WebSphere Application Server in the online help (click **Content tab** -> **How do I...** -> **Secure applications**, and other security-related topics).

7.10 e-commerce model

This section provides a model of an e-commerce system for the following:

- Model overview
- Use case
- Technical walkthrough

7.10.1 Model overview

In this section we will discuss use case and technical walkthrough models as they relate to the User-to-Online Buying Pattern.

7.10.1.1 Use case overview

Once you have captured your initial requirements, you can write the use cases for the system. A use case is the enumeration of steps performed to complete a specific task from start to finish.

Rather than concentrating on how the system functions, use cases describe how the system will be used by its users. Also, use cases describe what the system must do from the user's perspective, and what use cases the different actors (users) initiate. Use cases define the system boundaries by identifying external agents (actors).

Though more detailed than requirements, use cases are still quite informal. A use case should be a natural-language description, not pseudocode, and it should avoid any unnecessary assumptions about implementation.

The style and amount of detail you include depends on the novelty of the system and your familiarity with the problem domain. The level of detail can also vary from one use case to another. Some use cases might describe high-level interactions of real-world elements, while others might describe low-level interactions of actual system objects (when such objects are known). Try to avoid mixing levels of abstraction within a single use case. The contents of uses cases are not subject to any rigorous, programmable semantics. They may be less imprecise or even ambiguous, but they should be succinct, high level, and understandable and verifiable by the user.

The use case model is described by the following constructs, as seen in Figure 22 on page 129:

- Actors (name, description, status, subclass, superclass, and associations)
- Use cases (number, subject area, business event, name, overview, preconditions, description, associations, inputs, outputs, traceable to, usability index, and notes)
- Communication-associations between actors and uses cases
- Relationships between use cases (same as use case associations)
- Termination outcomes
- Conditions affecting termination outcomes
- Termination outcomes decision table
- Use case scenarios (number, termination outcome, description, and notes)
- Problem domain concept definitions
- System steps decision table
- Flow of events table
- System sequence diagram

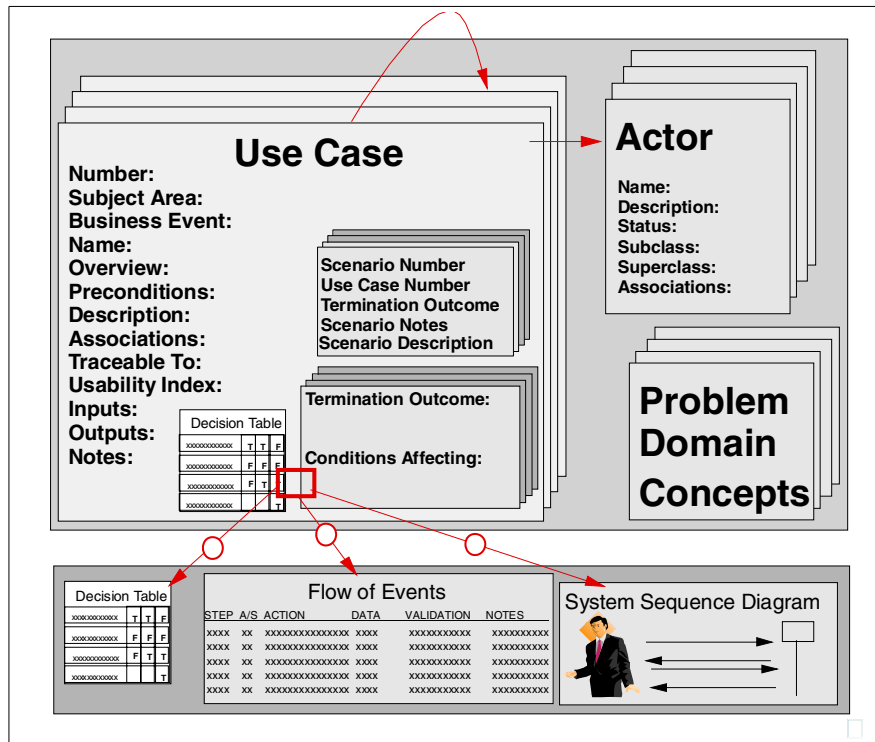


Figure 22. Use case model constructs

The value of using a use case model during development is that it uncovers the connections between the commerce application and the back-end systems. The use case model preserves the object-oriented model of the system, for further modifications and development.

Not adopting the use case model could yield more work during development. For example, a business process has changed that will require application modifications to support it. If you do not create a use case model, you run the risk that you have not identified hidden connections and interfaces. As a result, your project could require additional funding due to code rewrites, or delay the deployment of the application.

7.10.1.2 Technical walkthrough overview

A technical walkthrough deals with how the system works, concentrating on the infrastructure from end-to-end. It does not overly concern itself with the function of the application. Multiple walkthroughs may be developed for the

same logical architecture diagram, showing the operation of different functions and usage by people in different roles (shoppers, administrators).

7.10.1.3 Use case and technical walkthrough comparison

Although both use cases and technical walkthroughs describe the dynamic behavior of the system, the models take a different approach:

- The use cases describe what the interface is between the user and the system and focus more on business function, which defines the business requirements for the system.
- The technical walkthrough describes how the system works and focuses more on the end-to-end flow from the user interface device through the application and middleware components to the back-end systems.

7.10.2 Use case example

This example provides a very high-level description of an e-commerce system. The description below does not cover all the deliverables required from a use case. This is a sample of a whole use case work product.

7.10.2.1 Overall view - use case

Figure 23 provides an overview of an e-commerce system at the highest level, considering the functionality and actors.

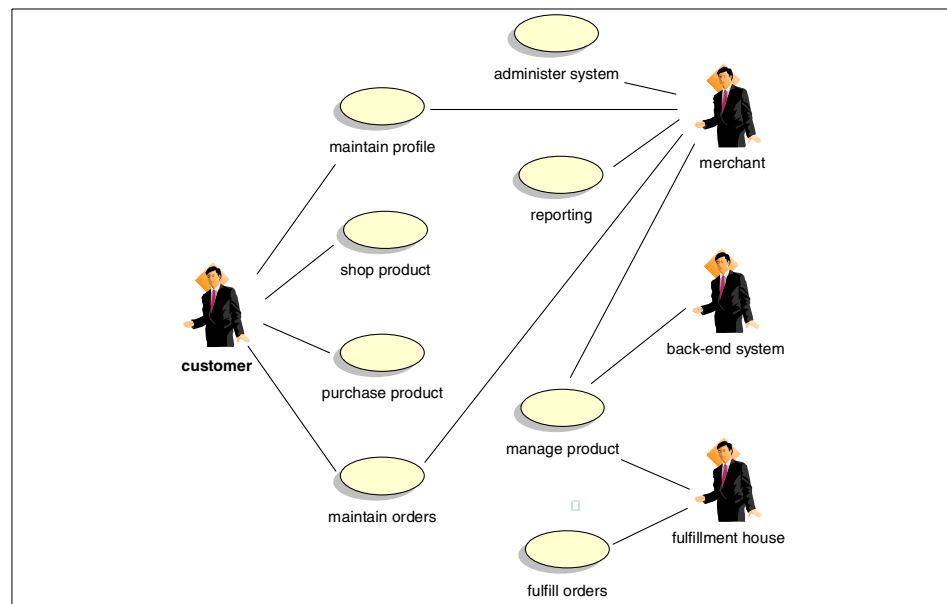


Figure 23. Overall view - use case

Actors

1. Customer

The party interested in buying merchandise on e-commerce site.

2. Merchant

Parties within the e-commerce company responsible for content, pricing and promotions.

3. Back-end system

Provides the content and product information for the front-end e-commerce application.

4. Fulfillment house

The party responsible for supplying and shipping merchandise ordered from the e-commerce site.

Use cases

1. Maintain profile

A customer is able to manage its profile, including all customer and merchandising-related information.

2. Shop for product

The customer is able to navigate and search through the product catalog, and maintain the shopping cart.

3. Purchase product

The customer identifies himself, commits to items in the shopping cart order, and processes checkout and credit card authorization.

4. Maintain orders

The customer is able to review orders, while the merchant is able to cancel or credit an order.

5. Administer system

The merchant is able to control aspects of the system including security, fraud detection rules, tax rates, shipping categories, etc.

6. Reporting

A business user wishes to report on aspects, settlements, orders, etc.

7. Manage product

The merchant, back-end system, and the fulfillment house have the rights to create and maintain catalogs and the product information including product prices, promotions, and fulfillment-specific information.

8. Fulfill orders

The fulfillment house responds to an order.

7.10.2.2 Manage product

The merchant, back-end system, and the fulfillment house have the right to create and maintain catalogs and the product information including product prices, promotions, and fulfillment-specific information, as seen in Figure 24.

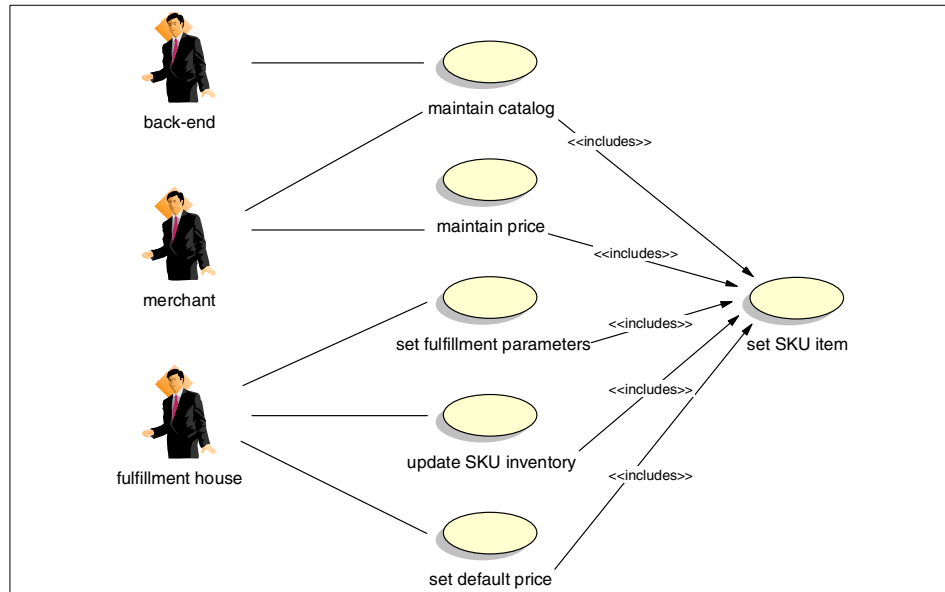


Figure 24. Manage product - use case

Actors

1. Back-end system

Provides the content and product information for the front-end e-commerce application.

2. Merchant

Parties within the e-commerce company responsible for content, pricing and promotions.

3. Fulfillment house

The party responsible for supplying and shipping merchandise ordered from the e-commerce site.

Use cases

1. Maintain catalog

The site administrator adds, removes and maintains presentation material.

2. Maintain price

The merchant is able to define price events, such as promotions, sales, enter product information, effective dates, prices, discounts, auction properties, advisor rules, etc.

3. Set fulfillment parameters

The user enters the fulfillment parameters relates to shipping.

4. Update SKU inventory

The fulfillment house is requested to provide inventory positions and supplies the information.

5. Set Default price

The fulfillment house supplies information on a product's cost and price.

7.10.2.3 Shop product

The customer is able to navigate and search through the product catalog, and maintain the shopping cart, as seen in Figure 25.

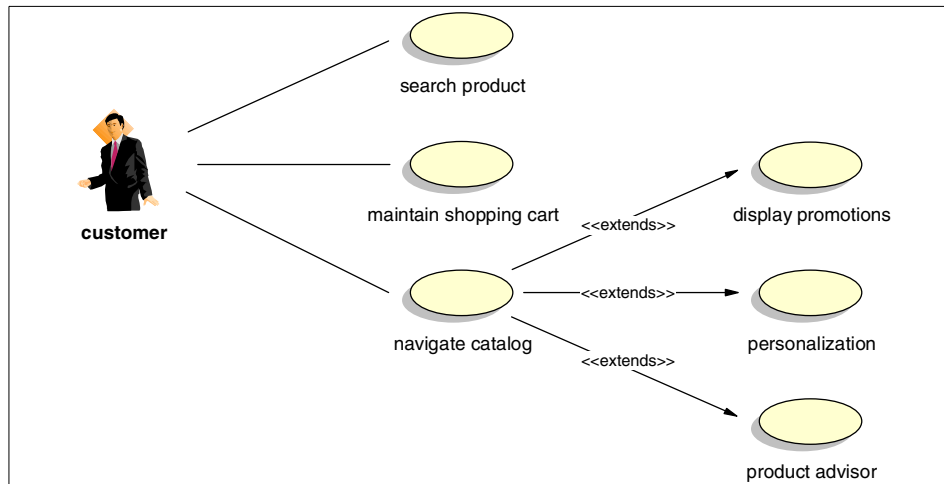


Figure 25. Shop product - use case

Actors

1. Customer

The party interested in buying merchandise on the e-commerce site.

Use cases

1. Search product

The customer provides search parameters for the desired product.

2. Maintain shopping cart

Customer decides to add, remove, review products in the shopping cart.

3. Navigate catalog

The customer selects an approach to browse products.

4. Display promotions

After the product selection the system displays any applicable promotions for that product.

5. Personalization

After the product selection the system looks up in the roles for any applicable product recommendations.

6. Product advisor

Other product recommendations, cheaper price, closest shipping, etc.

7.10.2.4 Purchase product

The customer identifies himself, commits the order in the shopping cart, and processes checkout and credit card authorization, as seen in Figure 26.

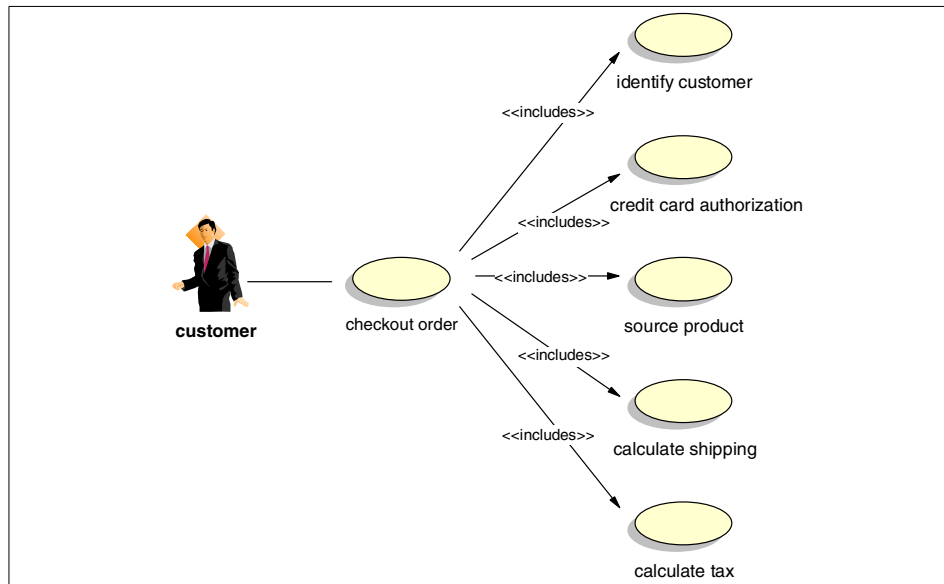


Figure 26. Purchase product - use case

Actors

1. Customer

The party interested in buying merchandise on the e-commerce site.

Use cases

1. Checkout order

The customer decides to purchase the items in the basket, enters credit card and shipping information, and approves the order.

2. Identify customers

The customer identifies himself with an identifier and password.

3. Credit card authorization

The customer request a credit card charge to be authorized, and enters the amount number, expiration, etc.

4. Source product

The customer requests a determination of which fulfillment house should supply an order item and provides the order item information.

5. Calculate shipping

The customer requests any shipping charges for an order to be calculated, and enters the order amount, ship from and ship to points.

6. Calculate tax

The customer requests any tax charges for an order to be calculated, and enters the order amount, ship from and ship to points.

7.10.2.5 Maintain orders

The customer is able to review orders, while the merchant is able to cancel or credit an order as seen in Figure 27.

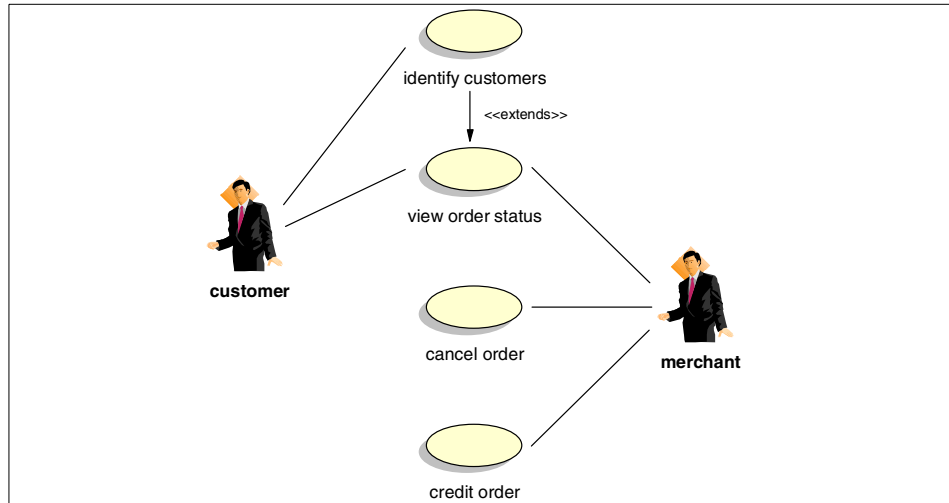


Figure 27. Maintain orders - use case

Actors

1. Customer

The party interested in buying merchandise on e-commerce site.

2. Merchant

Parties within the e-commerce company responsible for content, pricing and promotions.

Use cases

1. Identify customers

The customer identifies himself with an identifier and password.

2. View order status

The customer indicates he wishes to view the status of an order.

3. Cancel order

The customer indicates he wishes to cancel the order.

4. Credit order

The customer receives notices from the fulfillment house that he has returned a product and is due a credit for the item.

7.10.2.6 Fulfill orders

The fulfillment house responds to an order as seen in Figure 28.

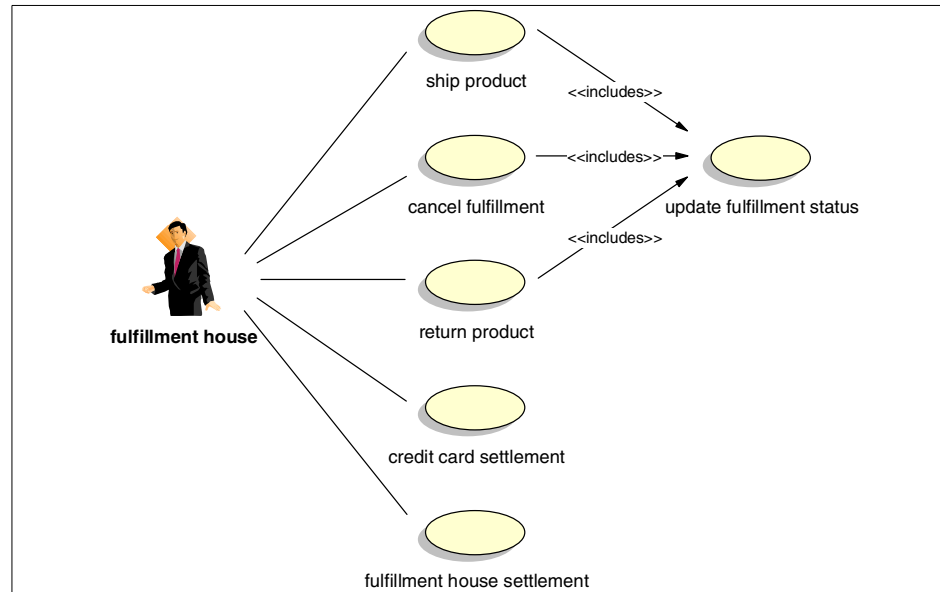


Figure 28. Fulfill orders - use case

Actors

1. Fulfillment house

The party responsible for supplying and shipping merchandise ordered from the e-commerce site.

Use cases

1. Ship product

The fulfillment house sends a notice of a shipping request.

2. Cancel fulfillment

The fulfillment house sends notice a of canceling a request.

3. Return product
The fulfillment house receives a returned product from the customer.
4. Credit card settlement
The fulfillment house sends notification of shipments to the e-commerce system.
5. Fulfillment house settlement
An invoice is received by the e-commerce system.
6. Update fulfillment status
The fulfillment house sends a notice that they have performed some task with an order.

7.10.3 Technical walkthrough example

The runtime topology, as seen in Figure 29 on page 138, shows the logical of a commerce Web site.

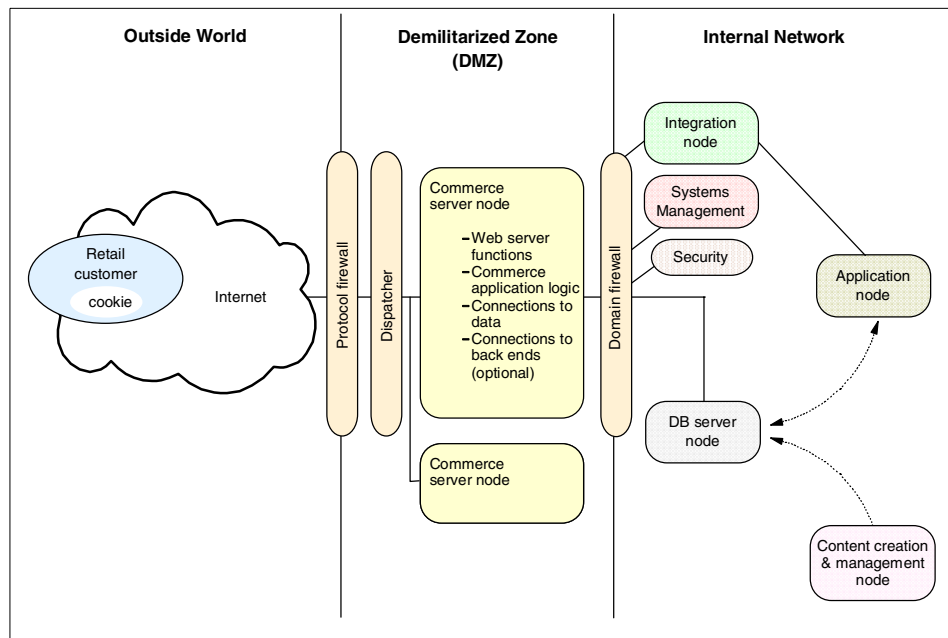


Figure 29. Runtime topology - logical nodes

7.10.3.1 Basic connection and first contact

1. Using a simple Web browser, the shopper accesses the Internet, and then chooses to link to the destination by typing in the URL.
2. The link provides a name that must be resolved into a specific network address. The address comes from either the local machine host directory or from the name server.
3. The message first is sent to the Protocol firewall, which is basically a filtering router (allows the HTTP and secure HTTP ports), then the message is passed to the network dispatcher.
4. The network dispatcher chooses which commerce server will respond to the message, based on load balancing and availability. Then the network dispatcher routes the message to the selected commerce server.
5. The commerce server serves the client. The first response should be the welcome page (index.html). Basically there is only one HTML file in the whole store, the index.html, and it redirects immediately to the main page, which is a Net.Data macro.

7.10.3.2 Site map

On the welcome page, or the main page, there is the navigation bar that provides navigation for the shopper to the main functions of the site.

- Welcome - the main page introduction
- Catalog - browsing the catalog
- Logon - logon as a registered user
- Register - register as a new shopper, or update the existing profile
- Address book - the user's address book
- Search - search in the catalog
- Interest list - interest list and shopping cart are the same in WebSphere Commerce Suite V4.1
- Order status - checking the status of the previous orders
- Customer service - customer service page, with contact information

Some pages are reachable only via the secure network connection. Each time the user accesses a secure document, the browser notifies the user.

7.10.3.3 Welcome and Customer Service Pages

The Welcome and the Customer Service pages should provide basic information. They may be customizable for other issues. For example, in a

Customer Service page there could be some customer relationship management (CRM) services.

7.10.3.4 User registration, logon

During the store creation there are different alternatives for managing users:

- No registration required
- Registration required
 - Register on entry
 - Register when ordering
- Optional registration

Note

In this example, we chose to require registration when ordering.

There may be differences between the elements in the list if the user is registered rather than just browsing. This decision is possible by the use of cookies. Cookies are set the first time the user accesses something unique to the session.

Registering

When a user registers, an entry is recorded into the user table with the information about the shopper.

1. The shopper clicks on the registration link.
2. The message is sent over the Internet through a firewall to the dispatcher, then to the selected commerce server. The connection is secure, using SSL.
3. The HTTP server gets the message, and recognize that this is a command: `/webapp/commerce/command/RegisterForm`, which belongs to the CGI application, so the HTTP server invokes the director.
4. The director accesses the commerce daemon, which sets the `REG_NEW` task (if the shopper is not logged in yet).
5. The task generates a blank registration page.
6. The page is sent by the commerce daemon, and served by the HTTP server.
7. The page is sent from the HTTP server though the Protocol firewall, over the Internet to the Web client browser.
8. The shopper fills in the mandatory fields.

Note

You should check that all mandatory fields are filled in with valid values. The best method for this is to validate the user input on the client side. This can be implemented by writing a JavaScript function, which is called by the OnSubmit event from the FROM element. See the JSP example in 15.4.7, “Integrating into the store” on page 373.

9. The shopper clicks the Submit button, then the form is sent with the POST method, and invokes the RegisterNew command on the server side.
10. The message is sent over the Internet through the firewall to the dispatcher, then to the selected commerce server.
11. The HTTP server invokes the director. The director calls the commerce daemon.
12. The command checks the required registration information parameters calling the AUDIT_REG task.
13. Then the command updates the registration record in the database, connecting to the database server node through the domain firewall.
14. The shopper login occurs, by creating a digitally signed cookie that is sent back to the browser.
15. On successful completion, a specified URL is called, in this example the renew.d2w Net.Data macro.
16. The macro is executed by the ExecMacro command, and the generated HTML page is sent back to the client.
17. The Web browser client displays the page and sets the cookie.

Register Update

In this case the only difference is that there is no need to create a new record in the database, because the original fields will be updated. The RegisterForm page sets the REG_UPDATE task. The RegisterUpdate command is used in place of RegisterNew, which invokes AUDIT_REG task, then updates the record.

Logon

When a shopper is already registered, there is no need to register again the next time he enters the store. There is a logon to access the secured pages, specific for the shopper logon ID.

1. The shopper clicks on the registration link.

2. The message is sent over the Internet through the firewall to the dispatcher, then to the selected commerce server. The connection is secure, using SSL.
3. The HTTP server gets the message, and recognize that this is a command: `/webapp/commerce/command/LogonForm`, which belongs to the CGI application, so the HTTP server invokes the director.
4. The director access the commerce daemon, which sets the LOGON_DSP task.
5. The task generates a blank logon page.
6. The page sent by the daemon, passes the director and is served by the HTTP server.
7. The is sent to the Web browser client via the Protocol firewall and Internet.
8. The shopper enters the shopper ID and password.
9. The shopper submits the form, then the form is sent out with the POST method, and invokes the Logon command on the server side.
10. The message is sent over the Internet through the firewall to the dispatcher, then to the selected commerce server.
11. The HTTP server invokes the director. The director calls the commerce daemon.
12. The command ensures that the shopper's shopper ID and password are valid.
13. Creates a digitally signed cookie that is sent back to identify the shopper's session.
14. Generates the page set by the URL parameter.

Note

You can specify any URL destination for the next page. This enables you to request a logon from any time, and go back to the original flow and continue the process.

15. The HTML page is sent back to the client via the Protocol firewall over the Internet.
16. The Web browser client displays the page and sets the cookie.

7.10.3.5 Browsing the catalog (JSP)

The catalog contains information about the products is the store for all products and item.

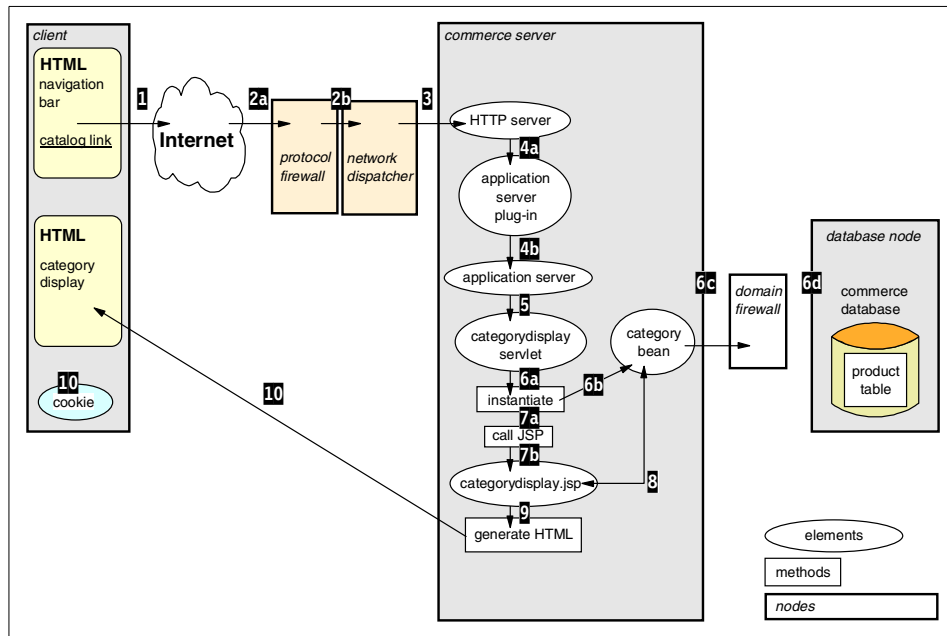


Figure 30. Category Display - technical walkthrough

Category and Product List Display

Category display is depicted in Figure 30 on page 143.

1. The shopper clicks the Catalog option.
2. The message is sent over the Internet through the firewall to the dispatcher, to the selected commerce server.
3. The HTTP server gets the request, and decides whether it is a CGI call or a servlet call depending on the URL. The /command/ path belongs to the CGI, while the /servlet/ path belongs to the servlet. In this example the catalog pages are JSPs.
4. In this case the HTTP server passes the request to the application server via the server plug-in.
5. The application server calls the CategoryDisplay servlet, this is equivalent to the CategoryDisplay command.
6. The servlet instantiates the CategoryBean, which accesses the commerce database and gets the data from the tables. The bean connects to the database node in the secure network via the domain firewall. The domain firewall allows database access on TCP/IP ports 50000 and 50001.

7. The servlet calls the `CategoryDisplay.jsp` which displays the required information. The application server compiles the `CategoryDisplay.jsp` into a servlet (first time, after then it is accessible as a servlet).
8. The compiled JSP access the `CategoryBean`, instantiated by the servlet, and uses it to get the properties, that belong to the category.
9. Finally, the compiled JSP generates the HTML page and sends it back to the Web browser client via the Internet.
10. The browser receives the HTML document and displays it to the shopper.
11. A cookie is set on the client's browser to identify the shopper during the session.

Note

A secure connection during catalog browsing is not necessary since the shopper does not have to register.

When the shopper reaches the lowest level of categories the next page that the `CategoryDisplay` servlet calls is the `ProductListDisplay.jsp`. The `ProductListDisplay.jsp` displays the products under the latest level of category, and links to the `ProductDisplay` command. The product list display uses the product list bean to access the commerce database.

Product Display

1. The shopper clicks on a product link.
2. The message is sent over the Internet through the firewall to the dispatcher, to the selected commerce server.
3. The HTTP server gets the request and passes the `ProductDisplay` servlet call to the application server via the server plug-in.
4. The application server calls the `ProductDisplay` servlet.
5. The servlet instantiates the `ProductBean` that accesses the commerce database and gets the data from the tables. The bean connects to the database node in the secure network via the domain firewall.
6. Then the servlet calls the `ProductDisplay.jsp` that displays the required information.
7. The application server compiles the `ProductDisplay.jsp` into a servlet (first time, after then it is accessible as a servlet).
8. The compiled JSP accesses the `ProductBean`, instantiated by the servlet, and uses it to get the properties that belong to the product.

9. Finally, the compiled JSP generates the HTML page, and sends it back to the Web browser client over the Internet.
10. The browser receives the HTML document and displays it to the shopper.

Note

There is an option to set up the store that requires that the user to be registered after entering the store.

7.10.3.6 Search the products

The shopper might want to search for products in the catalog when catalogs are very large.

1. The shopper clicks on the search link.
2. The message is sent over the Internet through the firewall to the dispatcher, then to the selected commerce server.
3. The message invokes the ExecMacro command and the Net.Data interpreter runs the macro, the search.d2w in our example, which generates the search page.
4. The page is sent back to the Web browser client over the Internet.
5. The shopper fills in the search field, and clicks the Search button.
6. The message is sent over the Internet through the firewall to the dispatcher, then to the selected commerce server.
7. The message invokes the ExecMacro command and the Net.Data interpreter runs the macro, the searchrslt.d2w in our example, which generates the Search Result page, with the products listed on the page.
8. The page is sent back to the Web browser client over the Internet.
9. Each item in the list on the Search Result page provides a link to the product page with the product information.

7.10.3.7 Add product to the shopping cart (interest list)

The shopper might want to set up an interest list, to put all interesting items found during catalog browsing, as seen in Figure 31 on page 146. The interest list behaves like a shopping cart. The shopper can check out the items to send an order.

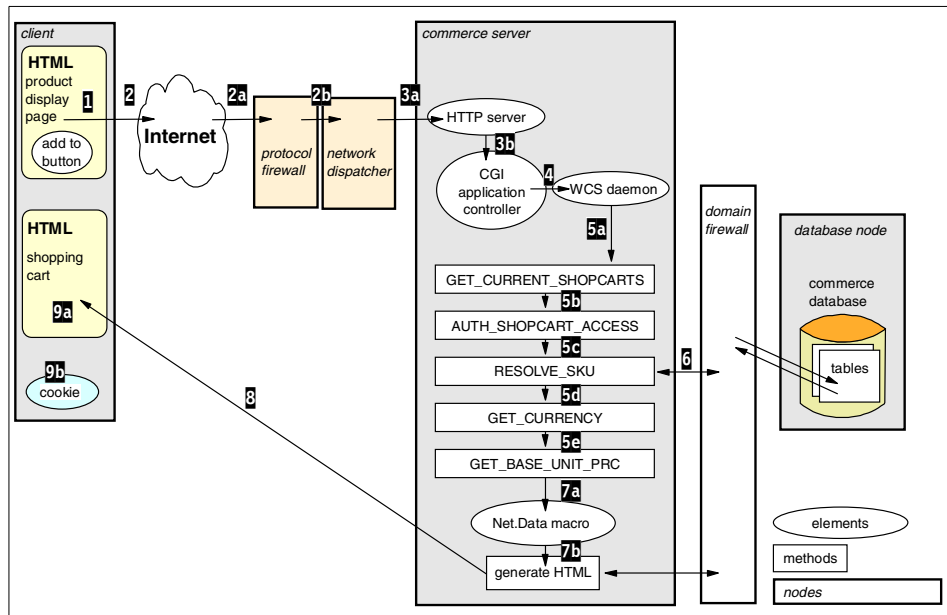


Figure 31. Add product to shopping cart - technological walkthrough

1. The first step is to find the right product. On the product page there is a button called Add to interest list.
2. As the shopper clicks the button the message is sent over the Internet through the firewall to the dispatcher, then to the selected commerce server.
3. The HTTP server get the message, and recognizes that this is a command: `/webapp/commerce/command/InterestItemAdd`, which belongs to the CGI application, so the HTTP server invokes the director.
4. The director calls the commerce daemon. The daemon calls the `Net.Data` macro to access the database. A commerce component will be invoked to create a shopping cart for the shopper, and to add a product to it. The shopping cart is stored on the database node. The shopping cart contains not only the products, selected by the shopper, but the shopper ID (if the shopper is registered), and the cookie to identify the session.
5. A `Net.Data` macro responsible for accessing the database node is executed and creates the shopping cart. Then it puts all the information in the cart. The following tasks are invoked:
 - a. `GET_CURRENT_SHOPCARTS` to determine the shopping cart.

- b. AUTH_SHOPCART_ACCESS to check for write permission for each of the specified shopping carts.
 - c. RESOLVE_SKU to determine the SKU for the products.
 - d. GET_CURRENCY to determine the currency to be used.
 - e. GET_BASE_UNIT_PRC to determine the price of the products.
6. The database component accesses and writes the data records into the tables on the database server node, and passes the information back to the calling Net.Data macro on the commerce server.
 7. A Net.Data macro generates the HTML page, and writes all the information from the database into the HTML page.
 8. The page is sent back to the Web browser client over the Internet.
 9. The Web browser client displays the page and refreshes the cookie.

Add more items to the shopping cart

This task differs from the previous one, because the shopping cart at this time already exists.

Note

The WebSphere Commerce Server can continue to be dispatched since there is no need to return to the same server as the last session because the session data and the shopping cart are stored on the database server.

In this case, the WebSphere Commerce Server is retrieving and updating the shopping cart instead of creating it as described previously.

7.10.3.8 Order check-out

When the shopper has added all the interested products to the shopping cart, it is time to check out an order and it send to the merchant.

1. The shopper clicks the Checkout button, on the Interest List page. This is a secure document, so the connection is secured with SSL.
2. The message is sent over the Internet through the firewall to the dispatcher, to the selected commerce server.
3. The HTTP server gets the message that contains the OrderItemList command, so the HTTP server invokes the director.
4. The director calls the commerce daemon.

5. The command sets the GET_CURRENT_SHOPCARTS process task to obtain the current shopping carts.
6. Then it sets the SHIPTO_LST view task to list all the shipping addresses that correspond to the shopper's order.
7. The view task generates the HTML page with the items, and a drop-down list for each item to choose the shipping address for the item.
8. The page is passed back to the daemon, which forwards the response to the controller.
9. The page is sent back to the Web browser client over the Internet.

Note

You can specify other parameters that enable you to check the inventory for the product.

10. When a shipping address is selected and the shopper clicks the Add to Shipping List, the OrderItemUpdate command is sent to the commerce server, which updates or creates a shipping record.
 - a. Calls the GET_BASE_SPE_PRC process task, gets the special price of the product item, and stores it in the SHIPTO table.
 - b. Calls the EXT_SHIPTO_PROC process task to perform additional processing to meet any unique requirements.
 - c. After all calls to the specified URL, in this example, it regenerates the shipping Addresses page with the updated information.
11. Then the page is sent to the Web browser client over the Internet.
12. When a shopper clicks the Continue button, the OrderItemDisplay command is sent to the commerce server.
13. The command lists all items for which the shopper has specified shipping addresses, by doing the following:
 - a. Calls the GET_CURRENCY process task to determine the currency to be used.
 - b. Calls the GET_BASE_SPE_PRC process task for each product and item to obtain their prices, and stores the values in the SHIPTO table.
14. On this page the shopper can specify the destination, set the quantity for the item, and can remove the item from the list. After any modifications the shopper needs to update the item by clicking the Update button.
15. The update message calls the OrderItemUpdate command, and the remove message calls the OrderItemDelete command. Both of the commands return

with the regenerated order list pages, and also make the modifications in the database.

16. When the shopper clicks the Submit button, the OrderDisplay command is executed to display the contents of the specified order by doing the following:
 - a. If needed, the GET_CUR_PENDING_ORDERS process task is invoked to obtain the current pending orders.
 - b. Calls the GET_CURRENCY process task to determine the currency to be used.
 - c. Calls the CHECK_INV process task for each product and item to ensure that there is enough quantity in stock.
 - d. Calls the GET_BASE_SPE_PRC process task for each product and item to obtain its price, and stores the values in the SHIPTO table.
 - e. For each suborder (the items in an order that are associated with a single shipping address):
 1. Creates a new suborder entry in the ORDERPAY table.
 2. Calls the GET_SUB_ORD_PROD_TOT process task to find the total price of all the products and items in the suborder.
 3. Calls the GET_SUB_ORD_PROD_TAX_TOT process task to find the total tax payable on the suborder.
 4. Calls the GET_SUB_ORD_SH_TOT process task to find the total shipping charges for the suborder.
 5. Updates the ORDERPAY table with the calculated amounts of the GET_SUB_ORD_PROD_TOT result, the GET_SUB_ORD_SH_TOT result, and the GET_SUB_ORD_PROD_TAX_TOT result.
 - f. Calls the GET_ORD_PROD_TOT process task to find the total price of all the products and items in the order.
 - g. Calls the GET_ORD_PROD_TAX_TOT process task to find the total tax payable on the order.
 - h. Calls the GET_ORD_SH_TOT process task to find the total shipping charges for the order.
 - i. Creates or updates the entry in the ORDERS table with the calculated amounts of the GET_ORD_PROD_TOT result, the GET_ORD_PROD_TAX_TOT result, and the GET_ORD_SH_TOT result.
 - j. Locks the order by setting the value.
 - k. Sets the ORD_DSP_PEN view task to display the pending order.

17. Then the generated page is sent to the Web browser client that displays the order.
18. In our example only the offline payment is possible in the store, so the shopper clicks the Pay offline button, and the OrderProcess command is sent to the commerce server.
19. The command ensures that the shopper invoking the command owns the order, and also that the order is pending and locked.
20. Updates the order to reflect the right information.
21. Determines the type of the payment (offline here).
22. Calls the UPDATE_INV process task to update the inventory for each product and item.
23. Calls the DO_PAYMENT_OFFLINE process task.
24. Calls the EXT_ORDER_PROC process task to perform additional processing that is needed for any unique requirements. In our example this task handles the back-end connection with the inventory management, by doing the following:
 - a. The XML message is generated and sent by the commerce server to the integration node via the domain firewall. The message is handled by MQSeries, which has a client on the commerce server, and a server on the integration node.
 - b. The integration node then connects to the back-end system on the application node, and forwards the message.
 - c. Then the back-end system on the application node, sends back a message through the integration node via the domain firewall to the commerce server by using MQSeries messages in XML format.
 - d. Then the commerce server updates the inventory information in the database on the database node.
25. The ORD_OK view task is called as soon as the order is received, before the order has been authorized. The authorization is sent in the background, after the shopper is shown the ORD_OK page.
26. The order status is changed to completed (C).
27. The final order is sent to the Web browser client as an HTML page.

7.10.3.9 Check order status

The shopper might want to know about the status of the order sent to the merchant. It is possible to make a connection to any back-end system and get detailed information about the exact status of the order.

1. The shopper clicks the Order status link. This is a secure document, so the connection is secured with SSL.
2. The message is sent over the Internet through the firewall to the dispatcher, to the selected commerce server.
3. The HTTP server gets the message containing the OrderList command, so the HTTP server invokes the director.
4. The director calls the commerce daemon.
5. The command sets the ORD_LST_COM task (because the C parameter is passed to the command, so the list returns with the completed orders).
6. The orders list HTML page is then generated with the list of orders completed.
7. The page is sent back to the Web browser client over the Internet.
8. The Web browser client displays the page.

7.10.3.10 The rules engine

The rules engine is responsible for recommending products to the shopper on the specified page, if there are any matching rule to the current circumstances.

There are two pages where the rules engine is accessed:

- Welcome page
- Interest List page

How it works:

1. While generating a page a view task is called. In the personalization you can enable the overridable functions (OFs) under the view task.
2. The task will attach the ND_RecommendProducts OFs to the selected view task.
3. The OFs invokes the recommend products rule service.
4. The rules engine find the right rules depending on the criteria and generates the response.
5. The response is presented to the shopper in an HTML page on the Web browser client.

7.11 Where to find more information

IBM Publications

- For Information on the IBM Application Framework for e-business:

<http://www.ibm.com/software/ebusiness>

- IBM Application Framework for e-business: Web Application Programming Model: <http://www.ibm.com/developer/features/framework/framework.html>
- WebSphere Commerce Server Library:
<http://www.ibm.com/software/webservers/commerce/library.html>
- *Patterns for e-business: User-to-Business Patterns for Topology 1 and 2 using WebSphere Advanced Edition*
- *Servlet and JSP Programming with IBM WebSphere Studio and VisualAge for Java*
- Documentation for IBM WebSphere Application Server, including the product library (which includes IBM HTTP server documentation), hints and tips, white papers, and other support information can be found at:
<http://www.ibm.com/software/webservers/appserv/support.html>

Other Publications

- HTML:
<http://www.w3.org/MarkUp>
- JSP:
<http://java.sun.com/products/jsp/>.

Chapter 8. Application development guidelines

In this chapter we describe the development process used to build an e-business application from end-to-end. Also, we will discuss the usage of the tools for producing the development work products in each development phases. By using the IBM Application Framework for e-business's rich set of components, we can reduce the effort and cycle time for developing Web applications. This chapter is organized into the following sections:

- Development process overview
- Solution outline
- Macro design
- Micro design
- Build cycle
- Deployment
- Organization
- e-commerce application functionality
- Development environment
- Where to find more information

8.1 Development process overview

Today it is quite common in the industry to develop object-oriented software using an iterative and incremental process. There are many defined methodologies used in the industry for the development process. Different companies typically adopt a recognized process and evolve it for their environment. For example, IBM Global Services (IGS) has its own development process for customer engagements.

The development process we discuss throughout this chapter is a simplified version of the IBM Global Services methodology. The process is divided into different phases. Each phase is done in a sequential manner and is subdivided into smaller phases. Some phases are run through only once. Others are done over and over again, forming the iterative and incremental part of the development process. The actual process and which phases you use may differ slightly depending on the development team or organization that uses the processes.

We have divided the whole development process into the following phases:

- Solution outline
- Macro design
- Micro design
- Build cycle
- Deployment

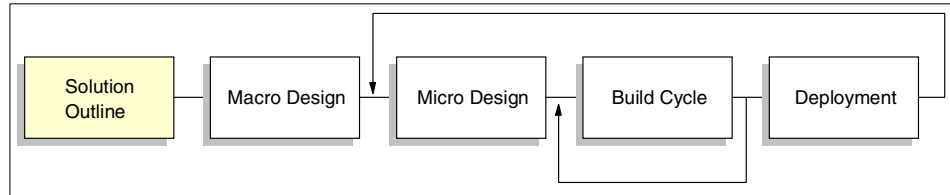


Figure 32. Development process overview

In the solution outline phase we decide the scope of the project, explore the essential business needs, develop an idea for the base architecture, and get a commitment from the project sponsor to start.

Next we start with the macro design that concentrates on gathering the detailed requirements, business process modeling, high-level analysis and design, the base architecture, and a plan for the following development phases, including a development release plan. The solution outline and macro design are usually done only once in a project.

Now the iterative and incremental part of the development starts. For each release of the developed e-business application, the micro design, build cycle, and deployment phases are completed. Usually, a certain set of use cases that need to be developed to meet a part of the system requirements make up a release. The releases are defined in the project plan produced in the previous phase. A release can be an internal one that is not deployed to any users. This is quite common for the early stages of big projects. Others, such as alpha or beta releases, might be deployed to a certain number of test users. It may take several iterations until a first official release of the application is deployed to the users. In turn, there are often several releases to the users until all requirements are met, plus maintenance releases to fix errors and other defects.

8.1.1 Application and architecture domains

A domain is a logical grouping of related work products. Different roles and skills fit into a domain through enabling specialization. Domains build the basis for method tailoring which is an important part of the IBM Global Services methodology.

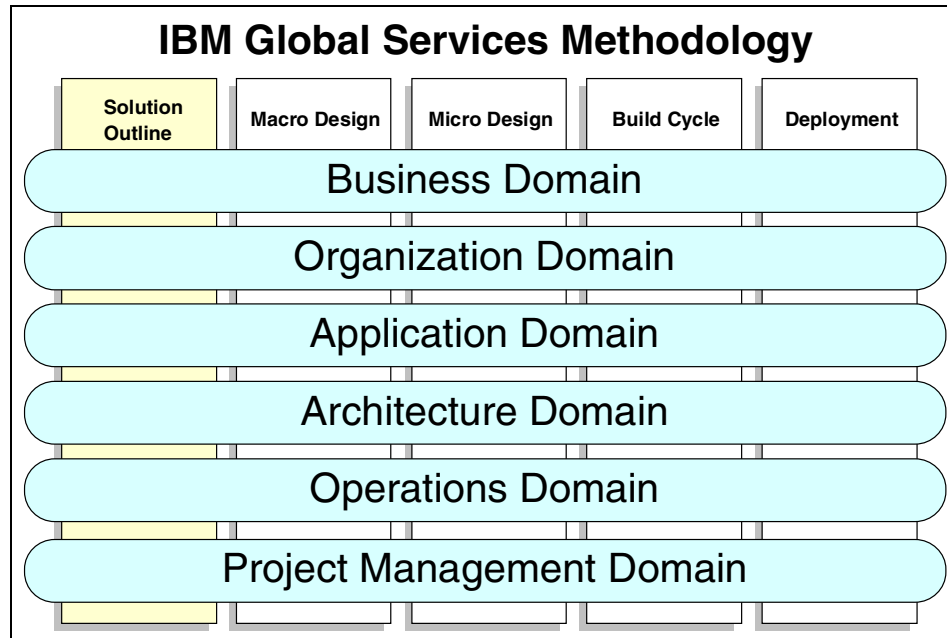


Figure 33. Domain concept in IBM Global Services methodology

Figure 33 on page 155 shows how the domains span the whole development process. There are six top-level domains. We will concentrate on the application and architecture domains. This does not mean that you should not be concerned with the other domains. For example, the use case model work product in the application domain is dependent on the business process model work product from the business domain.

8.2 Solution outline

The first phase of a development project is the startup. This phase normally begins with a small team of domain experts, analysts, and I/T architects exploring the requirements for the new solution. It is important to explore the existing environment to find out how the new application can fit. The target audience for the solution needs to be determined.

Based on all of the initial information an architecture for the application has to be determined. The team has to decide about the overall strategy of the solution that will drive the whole project. The strategy is based on the business impact the solution will have on the organization.

Question

Are there any back-end systems in use by the customer?

There are two different base strategies, depending on the organization business domain and planned solution:

- Web-up

The premise is to very quickly enable Web-based buying without close integration with back-end systems.

- Enterprise-out

The idea is to extend an existing order processing system to a new Web-based buying channel. In this case, there will be very close integration and re-use of back-end systems.

Usually customers have a good working business model. During the planning process it is essential to take into consideration the existing business methods and workflow. However, e-business applications often provide a business transformation. When developing the application, there could be some part of the existing business that requires transformation. Moreover, there could be new functionality provided by the Web platform. In the best case, during the architectural design, the solution should use the existing resources of the customer. In addition, there could be some part of the system which is better not to change.

The architectural decisions made are a separate work product and should be well documented. The work product is used as input for the macro design logical application topology and runtime topology.

8.3 Macro design

In the macro design phase the project team is usually extended from the few domain experts, analysts and I/T architects working in the solution outline phase to a broader skill set. This team works on the refinement of the results from the solution outline phase, including the following:

- Refine the requirements to come up with the business process model by identifying and describing key business use cases.

- Evaluate various technology options available for the implementation. At this point we need to choose a set of technologies for the application development as described in Chapter 6, “Technology options” on page 77.
- At this stage it is important to plan the deployment model, which requires that the runtime topology be finalized, which is discussed in Chapter 3, “Choosing the runtime topology” on page 21. Since such decisions have an implication for the long-term system management of the deployed application, we urge you to consider the guidelines provided by Chapter 9, “System management and security guidelines” on page 187.
- Subsequently the logical runtime environment needs to be mapped to physical instantiations of the products, as described in Chapter 4, “Product mapping” on page 39.

As in the solution outline phase, we need to document the architectural decisions as work products as they will serve as input for the following release cycles.

Other tasks that need to be done in the macro design phase include:

- Design and plan the test cases for function verification test (FVT) and system verification test (SVT).
- Set up the development environment.
- Create a development plan for the following release cycles.

We can start with media design, GUI prototyping, and the information architecture. These activities help us understand what the exact business requirements are. There are different approaches for these tasks:

For Web-up:

- Start from an existing Web site as your prototype
- Start from an abstract application flow model using such techniques as use cases and technical walkthroughs

For Enterprise-out:

- Start from an existing back-end system
- Start from a static object model, usually taken from a back-end system

Traditional applications usually present only structured data. Web applications on the other hand need to combine structured data and unstructured information on the same page. For example, an e-commerce book store not only needs to show the books, descriptions, and store locations etc., but it also has to show reader reviews about the books and

bestselling statistics. The combination of structured and unstructured information within the same application presents some unique challenges.

The Web application has to follow the business logic. Conversely, the business logic or application logic is determined by:

- Web platform
- Application design
- Attempting to make the GUI as simple as possible for the user

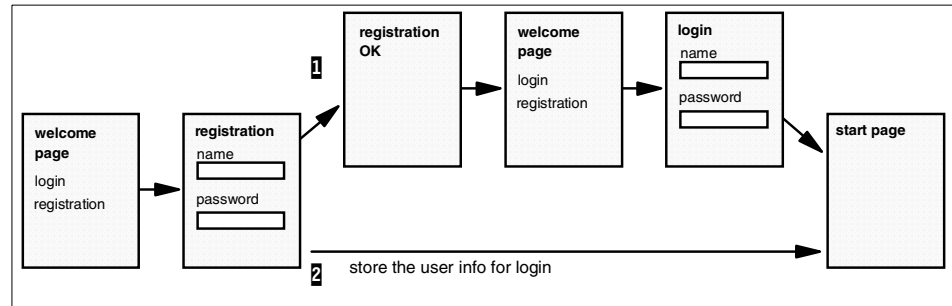


Figure 34. GUI design - registration example

For example, a simple user registration and login, as seen in Figure 34, could do the following:

1. The first approach follows every step, no intelligence, no foresight. The user has to go through five pages to reach the start page. Moreover, there are two forms.
2. The second case shortens the procedure. There are only two pages to reach the start page. The difference is that the application stores the necessary user information for the login.

This is a very simple example and we do not recommend the first approach, but there are many simple procedures in Web applications like this. Do not forget to look forward to make the application workflow simpler, faster, and traceable. The release cycles start after all architectural decisions has been made and documented.

8.4 Micro design

The development plan outlines several release cycles to implement the requirements iteratively and incrementally. After the macro design phase the

requirements of a project are captured in different work products such as the following:

- Business process model
- Domain use cases
- Domain class models
- Interaction diagrams for those use cases
- Technical walkthroughs

Each release cycle starts with the micro design that focuses on transforming the business model into a design model by taking the selected use cases and running them through a typical object-oriented development phase.

Transforming means that we use the business model to bring it to such a technically detailed level that it can be implemented.

During the micro design the lower level of the business functions are defined, so that the platform-dependent function description and sequence are ready to be implemented in the next cycle. The exact description, and the deep logical relations have to be recovered in this cycle; there are plenty of methods for this. These methods are not detailed in this book; only a high-level description of a e-commerce system can be found herein.

For more detail on use cases and technical walkthroughs refer to 7.10, “e-commerce model” on page 127.

8.5 Build cycle

In the release cycle, the micro design is followed by the build cycle. The designed system code is written, and tested in several build cycles. As in the micro design, each build cycle focuses only on the requirements valid for that particular release. So with every build cycle the developed system is growing in the functionality implemented. At this point we need to extend the source code terminology by including program code and markup language source, such as HTML and XML. The work products for the build cycle include the following:

- Write and unit test the source code
- Build the executable code if necessary (for example, all Java code)
- Perform various tests on the executable code and on the whole site
- Test the application in chosen runtime environment, including test, staging, and production

- Prepare for deployment

The incremental approach used to run the release cycles is also used for the different activities of the build cycle. Each iteration of the cycle transforms the design into tested executable code that is ready to be deployed.

8.5.1 Develop source code

We use the work products of the previous micro design phase as input for the coding phase.

8.5.1.1 Develop source code: work products

The *e-Commerce Patterns using WebSphere Commerce Suite, Patterns for e-business*, SG24-6156 application topologies described in this book suggest a layered approach, where the presentation logic and business logic are separated. We have divided the required work products for the developing the source code into the following categories:

1. Presentation elements

Presentation elements coordinate the application flow by accepting requests, invoking the required business logic, and invoking another presentation element to display the results. A JSP or Net.Data macro can be used to implement a presentation element, depending on the complexity of the interaction that is controlled. Presentation elements are implemented using different techniques. They reside on the Web server or Web application server, and are invoked from browser clients.

For static content, Web pages are built using HTML along with multimedia content, such as images, audio, and video files. Client-side scripting, such as JavaScript, can be incorporated into static pages to validate the values of the HTML forms used on Web browser client.

When the contents of a page change often, dynamic page construction techniques should be used. We call these view elements page constructors. JavaServer Pages (JSP) and Net.Data macros are used to construct dynamic contents on the WebSphere Commerce Suite server.

WebSphere Commerce Suite V4.1 provides a limited implementation of dynamic pages using JSPs. The use of Net.Data macros must still be used for most of the functionality provided in the WebSphere Commerce Suite. The three JSPs supported are as follows:

- Category display
- Product list display
- Product display

2. Business elements

Business elements or logic are implemented using JavaBeans or C++ modules, depending on the complexity and purpose of the particular model element.

Connectors are used to access data, local or remote databases, or other applications residing on the third tier. Connectors are Java class libraries that provide an application programming interface (API) with easy access to databases, middleware, or back-end systems.

3. Controller elements

Servlets are usually used to connect business logic and presentation elements. A layered approach will allow us to separate the business elements from the presentation elements, so that we only have to define the controllers. Both the presentation logic and the business logic need to agree with the controller.

4. Third-tier elements

There may be a need to add new components to the third tier to be able to access legacy systems or third-party applications. Also is possible that the back-end applications will need to be modified for integration with the newly developed Web application on the second tier. The work products needed to accomplish this integration task may differ very much depending on the actual system to be integrated. Some examples of the work products to accomplish an integration task are as follows:

- Batch program run on a back-end host
- Enabling database access to a host database
- MQSeries integration of back-end systems

8.5.1.2 Develop source code: process

The process of source code development includes the following activities:

- Writing the source code
- Transforming the source code to executable code
- Testing the written code
- Managing the source and the runtime code

These different activities are closely coupled and are usually executed in parallel.

Writing the source code

There are different input work products for the different types of code to be developed in this phase. GUI prototypes may have been produced to better understand the requirements and their possible solutions. Typically there will be a set of static content such as HTML pages, Net.Data macros, Java Servlets and JSPs. The prototype relies on the commerce engine for such features as user navigation, catalog and order management.

The static content (HTML pages) that serves as input pages needs to be separated from the dynamic content. The dynamic content, for example the result of a complex query, needs to be written as Net.Data macros or JSPs (not all functions available).

Testing the source code

Each development element should be unit tested by the developer who created and owns the source code. Since some components depend on input from other components (for example, a JSP needs the bean that is used to display products), it is very important to write test code that simulates the missing components.

Creating runtime code

The developed work products differ in the way they are deployed. All view elements, such as HTML, images, and JSPs, are deployed as is. There is no difference between source code and runtime code, because they are interpreted or compiled at runtime. All Java source code needs to be compiled into bytecode before deployment. It is important that the script and business logic developers plan for implementing their code within the build cycle and runtime environment.

Managing the source code

When developing source code with a team it is crucial to have good source code management. This issue becomes even more important when examining the many different types of technology that are used for developing an e-business application.

Some key features to look for when evaluating a source code management tool are its ability to share resources with other users in a team concept, integration with other tools, and the support of a version control system.

8.5.1.3 Develop source code: tools

During this phase the script and business logic developers generate the initial source code for the servlets, beans, and other Java classes using VisualAge for Java. Once logic is coded using VisualAge for Java, the class files can be imported into the WebSphere Commerce Studio by using a special built-in

interface for VisualAge for Java. From then on, the developers can use the same built-in tool to keep the Java classes in both tools in sync.

The view developer can import the static GUI prototypes into WebSphere Commerce Studio and from then on manages the code from the workbench. The view and script developers use the WebSphere Commerce Studio Page Designer to import code and to create new view code.

WebSphere Commerce Studio is also used to publish the developed code to the Web and application server, for testing, and finally into production. The Web server may have been used to test and evaluate the GUI prototypes in the early stages of the project. In addition, function verification testing can be performed on the Web application.

For more detailed information about the development tools refer to:

- 8.9.2, “Development tools” on page 170
- *Developing an e-business Application for the IBM WebSphere Application Server*, SG24-5423
- *Servlet and JSP Programming with IBM WebSphere Studio and VisualAge for Java*, SG24-5755

8.6 Deployment

Depending on the size of the components being developed during the build cycle, the development team may decide to build work products in several iterations. Each build cycle produces executable code. However, only the final build is deployed. It is up to the development and test team to decide which release of the build will be deployed.

8.6.1 Deployment: work products

A deployment plan should include the following:

- Project schedule for work products.
- Documented procedure on how to install and set up the required hardware and software.
- Documented procedure on how to install and set up the Web application.
- Create a systems management plan including security, availability, and recovery. This plan needs to be implemented before you deploy the application into a production environment. Detailed information on systems management can be found in Chapter 9, “System management and security guidelines” on page 187.

8.6.2 Deployment: process

When preparing for the deployment of a Web application, the plan for the hardware and software setup needs to be executed. This includes:

- Network routers and other networking components
- Firewall systems
- Load distribution systems
- Web application servers
- Database servers
- Integration nodes
- Back-end systems

8.6.3 Deployment: tools

In WebSphere Application Server, the Web content such as HTML, JSPs, GIFs are published to a set of directories while servlets, beans, and enterprise beans are published to a different set of directories. The structure for publishing the Web content may be dictated by local site rules. For example, there may be rules that say all content must be published to a single directory, all content for each subsite must be published to a single directory, or that content must be published to directories by type (HTML, images, etc.). The structure is not always the best way to organize the content for authoring (inside WebSphere Commerce Studio, for example). If the Web server is separated from the application server, static and dynamic contents are published to different servers. Finally, for scalability, it is often desirable to publish some static content, such as images, to a separate server from the HTML and logic.

There are many options in setting up and configuring the Web application server. Deploying the dynamic content to the WebSphere Application Server and configuring the deployed content is a task that needs to be planned and executed very carefully.

The WebSphere Commerce Studio workbench provides the ability to define an arbitrary directory structure to organize your e-business application's assets, view the relationships between the assets, and define how the assets will be published. Each asset is assigned a publishing target by default, mirroring the author directory structure. WebSphere Commerce Studio provides the ability to override this behavior and publish individual assets or entire folders to different directories, and often between multiple servers. The workbench adjusts the links to ensure that no matter how you publish your site, the links work properly.

8.7 Organization role

An e-business application is developed by a multi-disciplinary team. The skills include graphic artists, Web page designers, client and server-side script programmer, Java programmers, and traditionally skilled programmers. Depending on the size of the e-business application, there may be multiple team members for each skill or multiple roles performed by one developer.

A number of different types of skills and tools are required to implement various parts of a Web application. For example the skills and tools required to design an HTML page are vastly different from the skills and tools required to design and develop the business logic part of the application.

Whether there is only one person on the team or one hundred, the concept of the separation of roles and responsibilities is key to the successful development and maintenance of an e-business application.

- View developer

The view developer is responsible for the view part of the presentation logic, usually consisting of display pages written in HTML, JavaScript, and multimedia elements such as images, audio, and video. A developer performing this role also generates simple page constructors (Net.Data macros, and JSPs) using appropriate tools such as WebSphere Page Designer. The view developer may also be involved in GUI prototyping.

- Script developer

The script developer is responsible for creating complex dynamic pages. This role also includes the creation of part of the business logic related to the presentation logic that can be implemented using Net.Data macros, JSPs, or servlets. If servlets are used the developer needs Java programming skills.

The script developer depends on both the view and the business logic developer. The script developer writes the code that controls the interaction between user request, business logic execution, and invocation of the result display page.

- Business logic developer

The business logic developer is responsible for implementing the business logic, including data and third-tier access using skills in C++, Java, beans, enterprise beans, and connector programming.

The business logic developer needs to pay attention to the interface specifications of the code to ensure the script developer can safely call the business logic.

- Third-tier integration developer

If there are components to change or to add on the third tier, the third-tier integration developer is responsible for developing the required “glue” code to allow the integration of the new Web application with existing back-end systems or third-party applications. The skills needed can range from batch-job programming and database skills, to transaction application programming such as MQSeries.

For more information details on the new commerce functions provided, refer to the *WebSphere Commerce Suite v4.1 - What's New in WebSphere Commerce Suite* documentation.

8.8 e-Commerce application functionality

The WebSphere Commerce Suite Pro edition provides many built-in functions to create the commerce application. The application can be customized by using the WebSphere Commerce Studio, VisualAge for Java, and Page Designer.

Summary of new features available in WebSphere Commerce Suite V4.1:

Security

- Login/password
- LDAP

Catalog works

- Catalog browse
- Page preparation
 - JSP
 - Net.Data
- Full catalog search

Selling

- Product advisor
- Merchandising
 - Cross-sell
 - Up-selling
 - Accessories
 - Substitution

- Inventory - overstocked items
- Legal policy - product restrictions
- Time-based
- Affiliate programs
- Auctions - three built-in types (open-cry, sealed bid, dutch), but any can be developed
- Bonus points system
- Messaging system

Ordering

- Registration
- Quick order
- Multiple pending orders
- Multiple interest list
- Scheduled orders
- Order templates
- Re-order
- Order history tracking
- Checkout system
 - Billing
 - Shipping
 - Confirmation

Shopping cart

- Manage shopping cart
- Multiple shopping carts
- Quick view of shopping cart

Payment

- Multiple payment systems

Back-end integration

- MQSeries
- Connectors

For more detailed information, refer to the *WebSphere Commerce Suite V4.1 - What's New in WebSphere Commerce Suite* product documentation.

8.9 Development environment

This section contains information about the WebSphere Commerce Suite development environment. Currently, Windows NT 4.0 is the only platform provided that includes all the development tools. If your commerce site requires overridable functions and commands, a native C++ compiler is needed for the hardware platform and corresponding operating system.

The developing tools mainly exist only on Microsoft Windows platform, and most of them are only available just under NT 4.0 operating system. There are some other possibilities, but the only platform with all the tools is the Microsoft Windows NT 4.0.

8.9.1 IBM Application Framework for e-business

IBM has taken its e-business experience and developed a proven methodology for building e-business applications that are based on cross-platform technologies and open industry standards. The name given to this methodology is the IBM Application Framework for e-business, which is often referred to as the Framework.

The Framework is a productive environment that will help leverage the existing skills of your organization to deliver a new generation of scalable, robust Web applications that can be managed and migrated as needs change. The Framework allows you to do the following:

- Use your choice of tools
- Use your middleware, regardless of the vendor
- Build applications that can run on all IBM-supported operating systems

The IBM Application Framework e-business is very flexible. You have the ability to write applications on the operating system of your choice, while providing the application solution to the customer to test and deploy on the operating system of their choice as seen in Figure 35 on page 169.

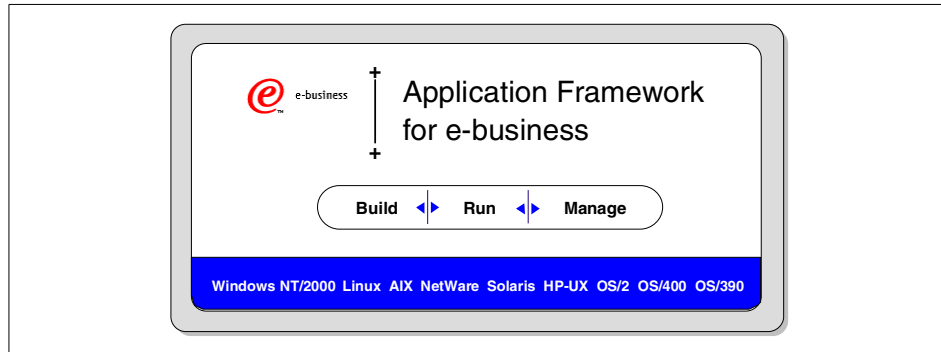


Figure 35. Application Framework for e-business

The Framework is based on industry standards including Java technology, XML and CORBA. Java technology's portability combined with the broad platform and service options provided by the Framework are unprecedented for deployment flexibility. For example, as your customer needs change, applications based on the Framework design principle can be moved from the initially deployed single Windows NT server, to multiple UNIX servers and then moved to a high-end S/390 without redesign or recoding. The Framework provides a model for high-output programming, while reducing the complexity and skill requirements for developing e-business applications. Your customers and your own investments in skill development and applications are protected by the wide support and adoption of Java technology by the I/T and software development industry.

The WebSphere Commerce Suite adopts the three pillars of the Framework: *build*, *manage*, and *run*. Figure 36 on page 170 highlights the development tools provided by WebSphere Commerce.

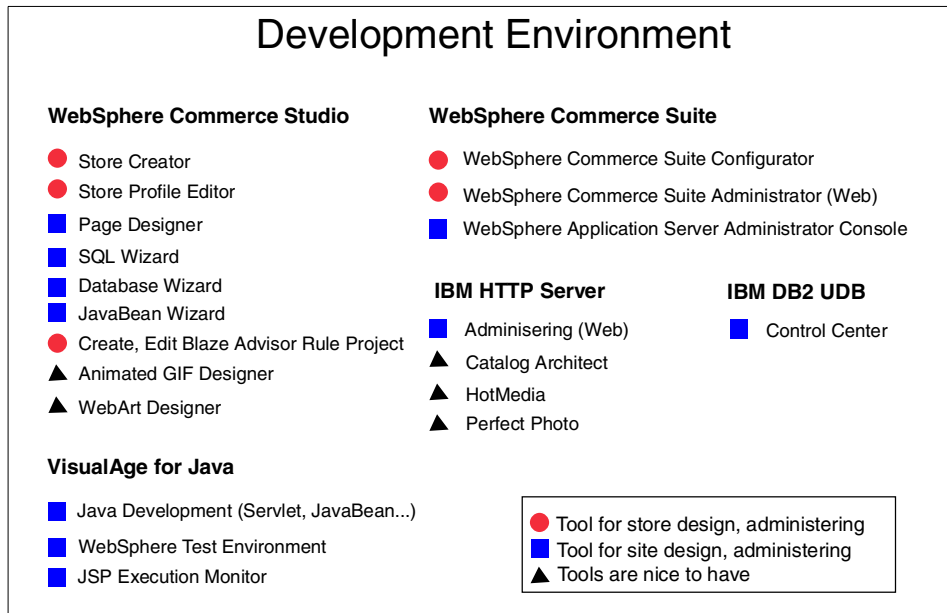


Figure 36. Development Environment

8.9.2 Development tools

The WebSphere Commerce Studio provides a wide pallet of development tools. There are some highly sophisticated designer environments and some wizard-driven simple tools. Figure 37 on page 171 provides a summary of applications and modules that come with WebSphere Commerce Studio. These tools can be used to develop the entire commerce store from scratch or modify an existing store.

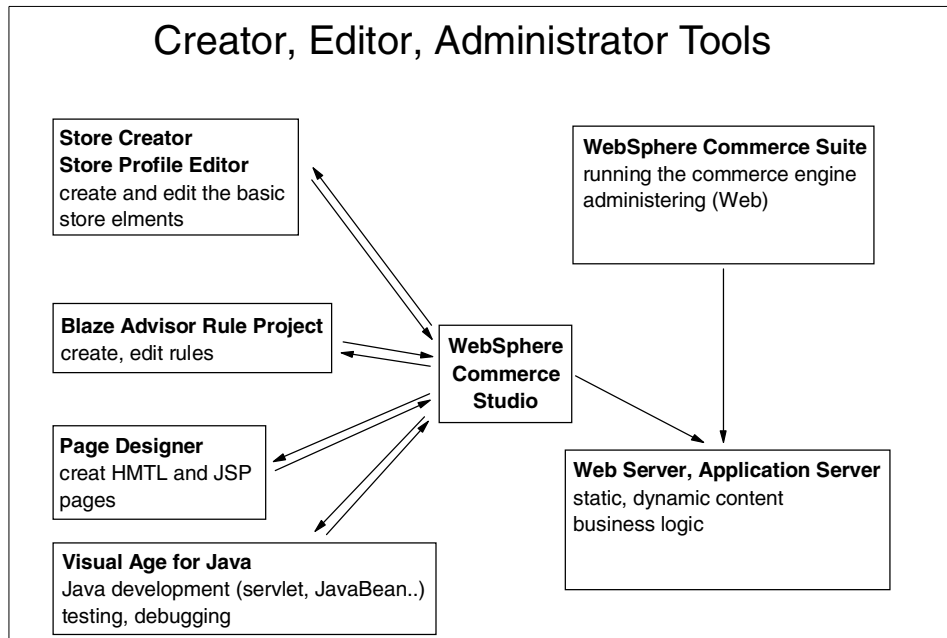


Figure 37. Tools in WebSphere Commerce Suite

8.9.2.1 WebSphere Commerce Studio

The WebSphere Commerce Studio, Professional Developer Edition includes the tools required to develop a store or mall. In addition to the development tools, a development license for the WebSphere Commerce Suite Pro Edition is included. As a result, WebSphere Commerce Studio, Developer Edition provides everything the store developer requires to create and test a store in a development environment.

The WebSphere Commerce Studio is a team-enabled environment, where the other development tools are integrated. It is used to organize the source code and runtime code of the entire development project and can be customized to launch a specific tool for any kind based on file type. The WebSphere Commerce Studio has very good publishing capabilities, making it easy to publish to different destinations, such as a test or production environment. Furthermore, it can be used to automatically publish changed files, or selectively publish parts of the files.

Developers use the WebSphere Commerce Studio workbench to create and modify all their assets, except for the work products created for the third tier. If multiple developers work on the project (as usual), these assets are stored on a shared file system or a version control system (VCS) sitting underneath

WebSphere Commerce Studio. Independent of which technology is being used for sharing files. The WebSphere Commerce Studio provides a check-out/check-in paradigm, where checked out assets are locked to avoid two users from simultaneously (unknowingly) updating the same asset.

While the shared file system is simple to set up, it does not provide the level of capabilities provided when using one of the VCSs. VCSs provide file versioning and are easier to manage with finer grained access control.

Note

Using a shared file system method requires that the project folder is shared and that the developers have the proper access.

The WebSphere Commerce Studio has the capability to integrate with any kind of development tool, so it is possible to have a special tool for each work product. Furthermore, WebSphere Commerce Studio comes with the WebSphere Page Designer and five integrated wizards as seen in Table 14.

Table 14. Wizards

Wizards	Usage
Database wizard	Create the input pages, servlets, beans, and JSP output pages for the database application.
SQL wizard	Create SQL queries, inserts, updates, and deletes.
JavaBean wizard	Create Web pages based on JavaBeans, for example, beans provided by a script or business logic developer. This wizard eases the work of the view developer when having to integrate any normal beans and the commands. Navigation and access beans are special beans produced with VisualAge for Java by the business logic developer.
Blaze Advisor	Create and edit the product recommendation rules.
Store Profile Editor	Edit the store profile.

For more detailed information on using WebSphere Studio, VisualAge for Java, and source code management, refer to *Servlet and JSP Programming with IBM WebSphere Studio and VisualAge for Java*, SG24-5755.

Store Creator

The Store Creator provides a head start in creating a store. It consists of a series of easy-to-use pages that guide you through 10 steps to create a basic store. You select a pre-designed shopping model upon which to base the

store, and then specify the store's appearance and page layout, contact information, currency, payment methods, sample products, sample taxes, and sample shipping providers. The Store Creator is suitable for the novice user or as a starting point to create a prototype store.

Create your own template

The Store Creator application uses an XML template file to provide you with the default options and components to create a store. With a little ingenuity, this file could be modified as you wish to build a customized template. The storecreator.xml file can be found in the <websphere_studio_path>\storecreator\properties directory. If you decide to experiment with this file, we suggest that you back up the original prior to modifying the file in the event that you would like to revert back to the original.

Store Profile Editor

The Store Profile Editor allows you to easily create or modify the store profile. The Store Profile Editor is composed of the following settings tabs:

- Preferences - allow you to create the store directory, and select or modify publishing options.
- Store Information - allows you to select or modify the store name, welcome message, default currency, and dual currency.
- Contact Information - allows you to add or modify the store's contact information.
- Catalog Templates - allows you to identify new category, product list, and product page templates for sample products.
- Layout - allows you to identify your store's home page, as well as any header, footer, and store logo files you wish to use.
- Tasks - allows you to enable tasks and create task pages for your store by associating them with the correct Net.Data macro.

Page Designer

WebSphere Commerce Studio Page Designer is an integrated tool used to create and maintain the pages for your Web site. It provides common page editing functions such as:

- WYSIWYG page editing (What you see is what you get)
- Browser preview of page
- Toolbar format features

- Drag and drop images and links into page
- Multiple views of page (Normal, HTML Source, Preview, Frame HTML Source, No Frames)

Page Designer supports HTML, Java Servlets, JavaBeans, JavaServer Pages (JSPs), JavaScript, and VBScript, and special editors to help you create and manage these elements. It can also be used to create JSPs or edit those you created with Studio wizards, add data from other beans and customize the result tables. You can also choose from a library of scripts and add dynamic functions to the Web pages without writing a line of code.

JSPs are well suited to creating category and product pages that display information from the catalog. Dynamic Web pages can be created by dragging and dropping WebSphere Commerce Suite specific beans to the Page Designer normal frame.

Create, edit Blaze Advisor Builder rule projects

The WebSphere Commerce Studio includes the Blaze Advisor Builder as a rule creation environment. This Java-based, graphical interface enables the site developer to create, and the business manager to edit, the rules that the site uses to make personalized product recommendations. Most tasks can be completed using either of the two methods.

After the rules project is created in the WebSphere Commerce Studio, use the Advisor Builder to create and extend all of the components for the rules project. The objects inside the Advisor Builder rules development environment display the Project Panel on the left side of the window. The tools are found in the toolbar at the top of the window, and output and messages are displayed across the bottom of the window. The central part of the screen is available as work space to be used with a variety of editors.

If the database schema has been modified, the Advisor Builder provides a tool that can be used to import the new database objects into the rule development environment. These new objects can then be included in the rules that are created for the Web site. It is recommended that only system administrators or the site developers use this tool.

Import wizard

Migrates existing sites to WebSphere Commerce Studio.

SQL wizard, Database wizard, JavaBean wizard

Add dynamic content to your Web pages by retrieving and updating information from common databases using server-side Java beans.

JAR wizard

Converts any Java class into a standard bean.

Animated GIF Designer, WebArt Designer

Page Designer uses WebArt Designer and Animated GIF Designer for creating and editing animated images.

8.9.2.2 VisualAge for Java

VisualAge for Java is an integrated visual environment that supports the complete cycle of Java program development. In particular, VisualAge for Java provides everything needed for development tasks described in this section.

VisualAge for Java is repository-based, uses incremental compiles, and has its own built-in version control management. VisualAge for Java is perfectly suited for team development, because it has a highly sophisticated management concept for code developed by a number of different users. The product comes with various Java connectors for all kinds of systems, ranging from JDBC for database access, CORBA or MQSeries for middleware, and IMS and CICS for back-end systems integration.

VisualAge for Java visual programming features provide developers a quick method for developing Java applets and applications. The *Visual Composition Editor* provides a point-and-click environment to do the following tasks:

- Design the user interface for your program
- Specify the behavior of the user interface elements
- Define the relationship between the user interface and the rest of the program

VisualAge for Java generates the Java code to implement what was visually selected in the Visual Composition Editor. In many cases complete programs are generated without writing any Java code.

In addition to its visual programming features, VisualAge for Java provides the following SmartGuides:

- Creating new applets.
- Creating new *program elements*. In VisualAge for Java, a program element is one of the following:
 - Project - the top-level program element in VisualAge for Java. A project contains packages.

- Package - the Java language construct. Packages contain classes and interfaces.
- Class - the Java language construct. Classes contain methods and fields.
- Interface - the Java language construct. Interfaces contain methods and fields. The fields in interfaces must be static final fields.
- Method - the Java language construct.
- Creating features for JavaBeans.
- Importing code from the file system and exporting code to the file system.

The script developer uses VisualAge for Java to edit, test, and debug the servlets and JSPs, and may also use the Java programming environment to monitor the HTML-generated output as the JSPs execute.

The business logic developer creates and edits classes, servlets, and beans inside the IDE, using SmartGuides or manually. The developer uses builders such as the Enterprise Access and Data Access Builder within VisualAge for Java to specify the details of the beans generated to access the legacy applications and data (exploiting the services provided by the WebSphere Application Server Advanced Edition connectors). With VisualAge for Java the business logic developer can edit, test, and debug all classes, servlets, and beans inside the development environment.

WebSphere Test Environment

While VisualAge for Java is a powerful servlet development environment, servlets only represent one architectural component of a Web application. A Web application also includes other resources such as HTML documents and JSP files. As stated earlier, the HTTPServer class only handles HTTP requests for servlets. It does not serve HTML documents or JSP files. Developing, testing, and debugging a Web application that incorporates all these components is a major challenge.

The WebSphere Test Environment is a version of the WebSphere Application Server that provides an execution environment for testing Web applications. In addition to supporting HTML requests for servlets, as is the case with the HTTPServer class, it serves both HTML documents and JSP files.

JSP execution monitor

The JSP Execution Monitor allows you to monitor the execution of JSP source, the JSP-generated Java source, and the HTML output. With the JSP Execution Monitor, you can efficiently monitor JSP run-time errors. The JSP

Execution Monitor displays the mapping between the JSP and its associated Java source code, and allows you to insert breakpoints in the JSP source.

If you find an error in a JSP page, you can also modify the JSP source in a text editor, and then run the JSP source in the JSP Execution Monitor. To load the updated version of the JSP source into the JSP Execution Monitor, you simply have to refresh your Web browser.

The JSP Execution Monitor highlights the location of syntax errors in both the JSP and JSP-generated Java source.

8.9.2.3 HotMedia

The IBM HotMedia assembly tool takes multiple images and other media types and assembles them into a HotMedia file. The HotMedia assembly tool does not replace content-creation tools. It takes existing media and concatenates them into a HotMedia file. This HotMedia file can then be played on a Web page with the supplied Java players.

8.9.2.4 Perfect Photo

IBM Perfect Photo allows you to enhance and manipulate digital images (for example, pictures taken from a digital camera or scanned images) for your Web site. Although this product is not automatically installed with Studio, it is included with Commerce Studio and can be used to manipulate images for your store and catalog.

8.9.2.5 C++ compiler

Overridable functions (OFs) in WebSphere Commerce Suite are written in C++ and require a platform-specific compiler. These development tools are not shipped with WebSphere Commerce Studio. Table 15 on page 177 provides a list of C++ compilers required.

Table 15. C++ compilers

Operating System	C++ compiler
IBM AIX 4.3	IBM VisualAge C++ 3.6.4
Microsoft Windows NT 4.0	Microsoft Visual C++ 6.0

For information on other platforms and the latest information on this topic, refer to <http://www.ibm.com/software/webservers/commerce/servers/versions.html/>.

8.9.3 WebSphere Commerce Suite Administration

WebSphere Commerce Suite is a bundled software package providing a wide scale of software products that enable a Web site to build and run e-commerce application.

8.9.3.1 WebSphere Commerce Suite Configuration Manager

The Configuration Manager allows you to perform administration and configuration tasks without having to deal with syntax-sensitive configuration files. Using this tool, you can perform the following functions:

- Create or delete a Commerce Suite instance
- Stop and start a Commerce Suite instance
- Change the configuration settings for a Commerce Suite instance
- Configure the database to serve as a staging server
- Enable or disable basic and advanced caching
- Change the rule server configuration
- Enable or disable the rule server

8.9.3.2 WebSphere Commerce Suite Administrator

The Commerce Suite Administrator consists of the following components that are used for the creation and administration of stores and malls:

- Site Manager
- Store Manager
- Product Advisor
- Samples

Site Manager

The Site Manager is a collection of online forms that you use to manage some high-level functions for electronic commerce. Use the Site Manager to establish a site, regardless of whether it is a single store or a mall. The person who manages these functions is called the *site administrator*. The site administrator can use the Site Manager to perform tasks related to the following topics:

- Security
 - Site and store access levels
 - Command security
 - Password and user ID administration

- Currency, tax and shipping specifications
- Shopper information
- Site appearance and behavior
 - Header, footer, and home page assignment
 - Categories for stores in mall (if applicable)
 - Task scope to specify the consistency of tasks across multiple stores in the mall (if applicable)
- Store state
- Rule server administration
- Personalization information administration
 - Specification of rules for particular view tasks (tasks that display pages) for a store

Store Manager

The Store Manager provides a collection of online forms, that are used for store-specific management functions. The person who manages these functions is called the *store administrator*. The store administrator can use these forms to perform tasks related to the following topics:

- Store information
- Product categories
- Product information
- Discounts
- Shipping services
- Payment
- Shopper groups
- Customer information
- Order delivery
- Auctions

Product Advisor

The Product Advisor is included with the WebSphere Commerce Suite, Pro Edition. It provides advanced catalog functions that help create an interactive, intelligent catalog. Using Product Advisor, a catalog that contains the following elements can be created:

- Marketing information

For example, information that is typically provided to a customer by a sales professional.

- Technical information

For example, information that is typically provided to a customer by a product specialist or subject matter expert.

- *Shopping metaphors* to help customers find products that suit their needs. There are three shopping metaphors provided:

- Sales Assistance metaphor

This metaphor is used by shoppers who have little knowledge of a particular product category. Sales Assistance will guide the shopper to appropriate products by asking a series of questions that will narrow down the selection process and then suggest which products meet their criteria.

- Product Exploration metaphor

This metaphor is used by shoppers who have some product knowledge. It helps them narrow down their product search by selecting product features from a feature list and then providing the shopper with a list of products that meet the required specifications.

- Product Comparison metaphor

Once the product selection has been narrowed down to a list of possible products, the Product Comparison metaphor helps shoppers to do a side-by-side comparison of products.

The Product Advisor is launched from the WebSphere Commerce Suite Administrator. It consists of the following components:

- Catalog builder

Prepares the product data for use in the electronic catalog.

- Shopping metaphor builders

Shopping metaphor builders are provided to set up different presentations of your catalog data, including shopping metaphors, sales assistance, product exploration, and product comparison.

Samples

Mall sample is included with the WebSphere Commerce Suite, Pro Edition. The Metropolitan mall is a sample included with the WebSphere Commerce Suite, Pro Edition is not intended to be used as a base for a production system. Some Commerce Suite functions, such as messaging, will not work if

implemented in the sample. In addition, you must disable caching before you install the Metropolitan Mall.

The Metropolitan mall demonstrates a mall scenario that implements some of the basic features and functions of WebSphere Commerce Suite. You can include these features in a mall without having to do much customization, and use the sample to help you generate ideas for your business. The sample implements a wide variety of Commerce Suite features since it uses almost all of the default commands and overridable functions provided by the Commerce Suite system to display store pages and perform specific business processes, such as adding items to the shopping cart and processing an order. The mall comes complete with sample data, and when you choose to install the sample, the database is automatically populated with this data.

8.9.3.3 WebSphere Catalog Architect

The IBM WebSphere Catalog Architect is a tool for entering and maintaining product data for your e-commerce site. Catalog Architect provides tools that make it easy to manage catalog data for IBM WebSphere Commerce Suite, Version 4.1. Catalog Architect can be used to create new data or migrate existing data into Catalog Architect from other data sources, including existing WebSphere Commerce Suite databases.

Publishing data from Catalog Architect validates the data, and puts it in a form that can be transferred into WebSphere Commerce Suite and then to the Web site. The following benefits are provided when using Catalog Architect:

- Tools make it easy to enter large amounts of data, and provide easier and better maintenance than using WebSphere Commerce Suite alone.
- The re-use of common facts about related data reduces data entry, and improves data consistency.
- Multiple views of the data provide alternative ways to enter and modify data, and make it easier to understand and visualize relationships.

The features of Catalog Architect ensure that the data is:

- Consistent, because some information is generated automatically, and is shared across related objects.
- Correct, through the use of standard pick lists and validity checks.
- Complete, because all fields are visible at one time, and the system checks to be sure all required data is filled in.

8.9.3.4 WebSphere Application Server Administrator's Console

The Administrator Console in WebSphere Application Server provides the tools and features to allow an administrator to do the following:

- Install and configure resources (for example, servlets and EJBs).
- Assign security to resources.
- Ensure all applications are available.
- Grant or revoke user access (for example, create a new account for a new employee and add the person to particular user groups).
- Monitor server performance and load statistics.
- Clone application components for improved performance.
- Resolve problems by viewing trace and log files and debug applications.

8.9.3.5 IBM DB2 Universal Database

DB2 stores the commerce database for the WebSphere Commerce Suite.

Control Center

Use the control center to manage systems, DB2 Universal Database instances, subsystems, databases, and database objects, such as tables and views. Also the control center can be used to optimize queries and work with DB2 commands, jobs, and scripts.

8.9.3.6 IBM HTTP Server

IBM HTTP Server is a Web server based on the Apache Web server developed by the Apache Group (<http://www.apache.org>). IBM HTTP Server includes the following functions that not available in the standard Apache Web server offering:

- Support for SSL secure connections.
- Fast Response Cache Accelerator (Windows NT and AIX only).
- WebDAV to collaboratively edit and manage files on remote Web servers.
- Ability to produce dynamic content with FastCGI.

Administration

The HTTP Server Administration Server greatly simplifies the once-manual task of configuring the Web server. First select a server to configure. The Administration Server will prompt you for configuration values, which are written to a configuration file when you click **Submit**.

8.10 Commands, tasks, overridable functions

Commands and overridable functions (OFs) represent the basic building blocks of the WebSphere Commerce Suite. They are the components that allow you to extend and customize the base system. Typically, *commands* will call *tasks*, which will be mapped to *OFs*. Commands indirectly call OFs through tasks as seen in Figure 38 on page 183. They can also indirectly call other commands, and OFs can call other tasks. Then these components access the database. At a high level, a command represents a static business process that delegates a well-defined piece of business logic to tasks.

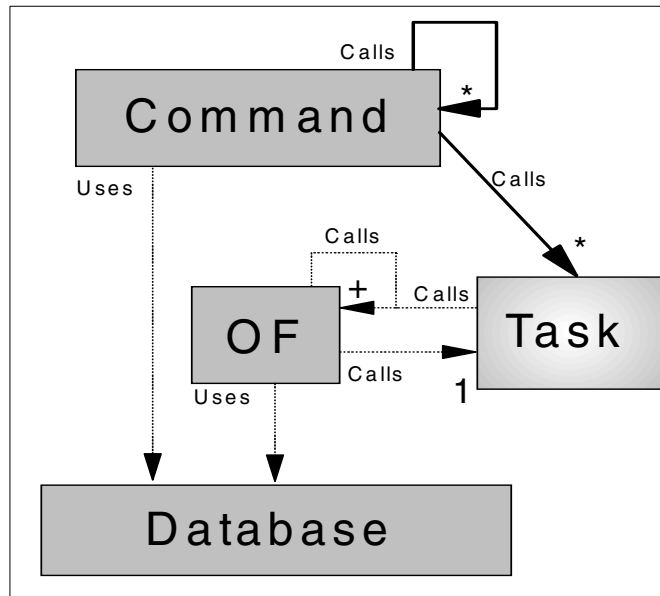


Figure 38. Commands, task, overridable functions - flow diagram

8.10.1 Commands

When designing a dynamic site such as a store, interaction from the user such as clicking on a URL will trigger some logic to be executed on the server. For example, a store allows a user to add a product to the shopping cart, and then view the order. With one click, the user will invoke the `OrderItemProcess` command, followed by the `OrderDisplay` command performed by a servlet, CGI program, or command. If the user reloads the page, or goes back to that page, or prints that page, then that same action will be triggered once again, adding the product to the order a second time before displaying

the page. In fact, any action that causes the browser to re-issue that URL will cause both steps to be executed.

This concept is the foundation for the difference between the two types of commands. Some commands have non repeatable side effects, whereas others do not. We generally associate commands that do not have these types of side effects to pages or views because these are the objects that the user can see and reload.

Process commands have non repeatable side effects, and are meant to return a redirection header to make the browser call another command. Display commands then end with a view page in case of success or an error page.

8.10.2 Tasks

WebSphere Commerce Suite provides process, view, and error tasks. Some tasks are meant to do processing work and are called PROCESS tasks. View and error tasks are meant to render/generate a page to be sent back to the browser. There is also another type of tasks, but we do not expose them fully. SYSTEM tasks are important to know about since they are invoked, and they are listed in the logs.

The only operation supported is the addition of new OFs to these tasks after (and not before) the currently assigned OFs. Do not attempt to remove or replace current OFs, or add OFs before the current OFs.

The last restriction is there to make sure that the OFs currently assigned to these tasks always get executed, no matter what. If you add an OF before these OFs, there is a possibility that your OF will fail, and therefore prevent the next OF in the task pipeline from executing.

8.10.3 Overridable functions

Overridable functions (OFs) are implementations associated with tasks. An OF will implement the semantics and interface of a task so that a command can call it. There is therefore a direct relationship between an OF and the task(s) it was written for. For an OF to be compatible with a task, it must obviously implement a behavior that is compatible with that specified by the task. The input parameters that the OF expects (set by the calling command) must be a subset of those defined by the task, and the output parameters returned by the OF must be a superset of those defined by the task (and therefore expected by the calling command). This means that you can write an OF that doesn't use some input parameters set by the command, and which would return more parameters than the command would expect back. This allows you to write super OFs that could be mapped to different tasks.

For instance, in WebSphere Commerce Suite, we have one OF that maps to all the product pricing tasks.

OFs associated with display tasks, whether error or view, are responsible for populating the HTTP request with a browser-viewable document. The WebSphere Commerce Suite has implemented two OFs that use Net.Data to render pages. One of these OFs takes in a file to render, while the other uses its invoking task and information in the database to get a filename back.

All other OFs are associated to PROCESS tasks, and are meant to perform some computational work such as checking an inventory for a product, computing a discount price, or invoking a payment subsystem. In the case of an error, most process OFs will invoke an error task. In general, OFs can invoke tasks the same way commands would.

8.11 Programming logic behind the application

The programming logic behind the application is comprised of business logic and presentation logic.

8.11.1 Business logic

Business logic provides the functional process for presentation. The HTML pages on the client side are generated by the server. The result pages are prepared by the presentation logic, and the presentation logic is driven by the business logic. OFs within WebSphere Commerce Suite provide access to the business logic. Business logic may be needed beyond what is provided with WebSphere Commerce Suite. The WebSphere Application Server provides access to JavaBeans and Enterprise JavaBeans technology.

8.11.2 Presentation logic

The presentation logic depends on the business processes. The connection between the business and the presentation logic are the controllers; within WebSphere Commerce Suite these controllers are the servlets. Technically JavaBeans and JSPs are connected via servlets.

8.12 Where to find more information

IBM Publications

- *Developing an e-business Application for the IBM WebSphere Application Server*, SG24-5423

- *Patterns for e-business: User-to-Business Patterns for Topology 1 and 2 using WebSphere Advanced Edition*, SG24-5864
- *IBM WebSphere Commerce Suite Pro Edition, Commands, Task, Overridable Functions and Database Tables V4.1*
- *Servlet and JSP Programming with IBM WebSphere Studio and VisualAge for Java*, SG24-5755
- *Design and Implement Servlets, JSPs, and EJBs for IBM WebSphere Application Server*, SG24-5754
- *WWW Programming: VisualAge for C++ and ST*, SG24-4734

Other Publications

- *Object-Oriented Analysis and Design with Applications*
- *Object-Oriented Software Engineering*
- *Object-Oriented Modeling and Design*

Chapter 9. System management and security guidelines

In this chapter we focus on the activities involved in systems management and security for the WebSphere Commerce Suite. This chapter is organized into the following sections:

- Systems management guidelines
- Systems management product guidelines
- Security guidelines
- Backup and recovery guidelines
- Where to find more information

9.1 Systems management guidelines

Once the application has been deployed in production the runtime environment and the application need to be managed. As part of the development process a systems management plan is created early in the planning cycle. We have categorized a set of post-implementation systems management activities:

- Application management
- Performance monitoring
- Availability management
- Security management
- Disaster recovery
- Operating system and network administration
- Asset management
- Software distribution
- Problem reporting
- Change management

Looking at this list of activities, system management is certainly not trivial. In fact, each of these activities requires highly specific skills and professional experience to perform them competently. Besides the skills factor, you will also have to decide on a set of tools to manage the system management activities.

Beyond the technical challenge of systems management, there is also the added pressure from management. In many situations, you will be bound by service level agreements (SLAs). Such agreements typically cover system availability hours, system utilization and problem resolution response time. These measurements will be collected, tabulated, and reviewed on a regular basis by management, to ensure accountability and a well-maintained system. Thus, you will also require reporting tools to facilitate the SLA review.

With all of the management focus and targets to meet, it makes systems management more daunting and challenging. Since it is important, it is our recommendation that you start planning early. Incorporate system management requirements in the early phases of your design, since the design will affect how you eventually manage it. Conversely, what tools are available to manage your system also affect your application design.

A detailed discussion of each topic is beyond the scope of this redbook. What we will focus on are the key system management activities related to the e-Commerce Patterns Using WebSphere Commerce Suite.

9.1.1 Managing your WebSphere Commerce Site

In the example seen in Figure 39 on page 188, the network is divided into three segments: an internal (or secure) network, a DMZ, and the public (or external) network. All external user access will have to flow through the external (or protocol) firewall to gain access to the WebSphere Commerce Server in the DMZ. Depending on the type of user request, the WebSphere Commerce Server may forward the request through the internal (or domain) firewall to the database server in the internal network. The application server will process the request and send any response back to the external customer through the protocol and domain firewalls.

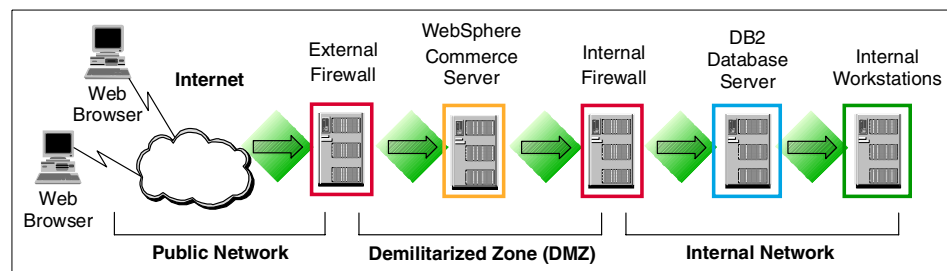


Figure 39. Managing WebSphere resources

The WebSphere Commerce Suite application is a combination of HTML pages, JSPs, servlets, Net.Data macros, and EJB resources. These

resources will be deployed and managed under the WebSphere Application Server and WebSphere Commerce server environment. In our scenario shown in Figure 39, the WebSphere Application Server and the WebSphere Commerce Server will host JSPs, servlets, Net.Data macros and EJBs. In the following discussion, we will refer to the WebSphere-specific terms in Table 16.

Table 16. Description of WebSphere terms

WebSphere Terms	Description
Web resources	Refers to servlets, JSPs, Net.Data macros and HTML pages.
Web application	Application consisting of Web resources.
Enterprise application	Application consisting of Web applications and EJBs.
Application server	A JVM runtime service that handles user requests from Web applications.

9.1.2 WebSphere resource management

WebSphere resources use core underlying services such as the servlet engine, EJB engines, security application, and cluster models residing in the application server. These services manage the Web resources and applications that are hosted by the WebSphere Commerce Server and the WebSphere Application Server. Depending on application requirements a stand-alone application server or multiple application servers with failover support capability can be used.

The WebSphere Application Server Administrative Console is used to manage, deploy, and configure the WebSphere resources. The WebSphere Commerce Suite Administrator provides system management capability to the Site Manager and Store Manager. For more information on the topic, refer to:

- *Application Server Solution Guide, Enterprise Edition: Getting Started, SG24-5320*
- *Patterns for e-business: User-to-Business Patterns for Topology 1 and 2 using WebSphere Advanced Edition, SG24-5864*
- *IBM WebSphere Commerce Suite Fundamentals, GC09-2994*
- *IBM WebSphere Commerce Suite Pro Edition, Commerce Suite Administrator V4.1*

9.2 Systems management product guidelines

This section provides information on systems management product guidelines.

9.2.1 WebSphere Application Server Administrator's Console

The WebSphere Application Server Administrator's Console is used to manage, deploy and configure the WebSphere resources.

9.2.1.1 Administrative tasks

This section discusses the general administrative tasks for the WebSphere Application Server.

Configuring applications and their components

The WebSphere Application Server provides support for servlets, beans, enterprise beans, JSPs, and Web pages that work together to perform a particular business logic function to create an enterprise application. After configuring an application, the Administrative Console can be used to start and stop the application as a logical unit. For example, when the application is started, all of the application components such as servlets, enterprise beans, etc. are started.

Controlling access to applications (security)

After configuring applications, we will likely want to limit access to them. For example, the public should not be permitted to use an application that accesses a database containing sensitive company information. The WebSphere Application Server can establish and enforce authentication, authorization, and delegation policies to control access to the applications.

Performing daily administrative operations

Day-to-day administration activities include the following:

- Ensuring that resources are available (running).
- Starting and stopping servers and servlets as necessary.
- Making incremental adjustments to the configurations of resources in the administrative domain.

Modifications can be small, such as granting permissions to access a new application, reinstalling an enterprise bean after changing a JAR file, or changing the frequency with which servers are queried to determine their state. Large-scale modifications can include introducing new resources into the domain or redefining the mix of resources in an application.

Analyzing usage statistics and performance

Resource analysis tools can be used to review current and historical information about resources in the domain. You can monitor performance and load statistics for servlets, enterprise beans, sessions, database connection pools, and server resources.

Optimizing performance

Resources in an administrative domain can be cloned to improve performance or availability. For example, application servers can be cloned to form a server group (a collection of identical instances of application server processes).

Cloning application servers improves the throughput of client remote-method invocations by distributing the load among the members of the group. It also improves availability and can prevent a single point of failure.

Cloning can be used to simplify configuration tasks. For example, you can configure a resource, test the configuration, and then duplicate the resource for use on other nodes in the domain.

Troubleshooting

Troubleshooting includes the following:

- Monitor transactions, forcing outcomes when necessary
- Analyze resources such as servlets and enterprise beans
- Trace and debug applications
- View traces, logs, and messages

9.2.1.2 Administrative terminology

The WebSphere Application Server provides administrators with a single-system view of applications and resources, such as servlets, that are typically deployed across multiple machines in a distributed environment. An administrator working on one physical machine can remotely administer resources located on another physical machine.

In the WebSphere administrative model:

- Node
A physical machine is called a *node*.
- Administrative server
Each node contains an *administrative server* for administering the resources on it.

- Administrative repository

Each administrative server stores its administrative data in an *administrative repository*, typically a DB2 database. The person installing IBM WebSphere Application Server specifies which administrative repository a given administrative server will use.

- Administrative domain

An *administrative domain* is a set of one or more nodes that administrative servers share an administrative repository. The administrative domain allows the nodes to be aware of one another and distribute applications among themselves.

- Topology

Each administrative domain has its own *topology*, comprised of the nodes in the administrative domain and the resources those nodes contain.

- Administrative resources

The resources on a node, such as servlets, class files, and enterprise bean JAR files, are represented as *administrative resources* in the administrative domain. An administrative resource, such as a servlet, holds configuration information about a resource, such as servlets installed on a node. It provides a way to start, stop, and otherwise manage the real resource, perhaps remotely.

- Containment hierarchy

The topology reveals the *containment hierarchy*, which is simply a tree illustrating how the administrative resources in the topology are related. The hierarchy shows which resources are parents or children of other resources. The Types tab of the WebSphere Administrative Console demonstrates the containment hierarchy.

- WebSphere Administrator's Console

The *WebSphere Administrator's Console* enables administrators to access the administrative server on each node in the administrative domain and provides a view of the domain's topology. It supplies task wizards for managing and combining resources in the topology.

9.2.1.3 Relationships among administered resources

Before discussing each of the administrative resources available in the administrative domain, it is important to understand the containment hierarchy that determines how the resources are interrelated.

The containment hierarchy imposes a structure on the topology of the administrative domain. Becoming familiar with the hierarchy will make your administrative tasks seem easier to accomplish.

If you have worked with directories and files, you have already been introduced to the concept of containment. Containment defines a relationship in which one resource is part of another resource.

If you have used programs with tree views, such as Microsoft Windows Explorer, you have literally seen containment hierarchies in action. From your experience with such file management programs, you probably realize that:

- Directories contain files
- Files are contained by directories

Using different terms:

- Directories are the parents of files
- Files are children of directories
- Files in the same directory are siblings

If you have ever tried placing a directory in a file, you know that your file management program will not allow it. The directory file containment hierarchy imposes the rule that directories can contain files, but files cannot contain directories.

Similar relationships, and rules, exist among resources in the WebSphere Application Server administrative domain. For example, in WebSphere Application Server, an application server resource contains a servlet engine and one or more EJB containers.

Note

The WebSphere Commerce Suite containment hierarchy allows only one servlet engine per application server.

The servlet engine contains one or more Web applications, each of which contains one or more servlet. The EJB container contains enterprise beans, of course. You cannot add a servlet to the administrative domain unless a Web application exists in which to place the servlet. Similarly, an application server and servlet engine must exist to support the Web application containing the servlets.

9.2.1.4 Resources administration

This section provides an overview of the administrative resources that can be administered using the WebSphere Administrative Console. Some administrative resources represent Java component files on your system such as servlets and enterprise beans. Other administrative resources provide support for managing, combining, distributing, and securing these components resources such as session managers, OLT controllers, and servlet redirectors.

Although the administrative domain is comprised of resources, the WebSphere Administrative Console provides task wizards. You can take a task-oriented approach or a resource-oriented approach, or both -- whatever you prefer.

Administrative resources found in the administrative domain containment hierarchy include:

- Nodes
- Servlets
- Web Applications
- Servlet Engine
- Application Servers
- Enterprise Beans
- EJB Containers
- Enterprise Applications
- Data Sources
- JDBC Drivers
- Models
- Servlet Redirectors
- Virtual Hosts
- Web Resources
- Session Managers
- User Profiles
- OLT Controllers
- Generic Servers

9.2.1.5 Configuring default values

The previous section discussed the many types of resources that can be configured in an administrative domain. When configuring the resources we recommend that you review the following:

- Each resource has custom and default properties.
- Some configuration changes do not take effect immediately, and can be performed in batches.
- Most properties are not required.

9.2.1.6 Default configuration

The IBM WebSphere Application Server V3.02 installation program provides a *default administrative configuration* option that populates the administrative domain with administrative resources with default settings.

Within the containment hierarchy, most administrative resources have a required parent. The default configuration provides many of these parent containers to support servlets and enterprise beans.

For example, in the administrative domain:

1. A servlet must be contained by a Web application.
2. A Web application must be contained by a servlet engine.
3. A servlet engine must be contained by an application server.
4. An application server must be contained by a node.

Without the default configuration, you would need to configure application server, servlet engine, and Web application resources before configuring a single servlet.

With the default configuration, you can immediately configure your servlet. You can specify that it will be contained in the default application server, servlet engine, and Web application.

The default configuration includes an application server (default server), container (default container), application (default app), and so on. These resources can be found in the topology tree on the Topology tab.

If you find that the default configuration was not installed, see the *WebSphere Application Server Advanced Edition, Getting Started V3.02* product guide for a discussion of your options for acquiring the default configuration. If you are inexperienced with the administrative model, it is highly recommended that you install and use the default administrative configuration in your

development environment, and in your production environment if it suits your needs.

9.2.1.7 Centralized administration

The WebSphere Application Server provides centralized administration of application servers, servlets, and other resources. An administrative server tracks a domain's contents and activities by maintaining an administrative repository. The repository is the database of information about an administrative domain and can be shared by several administrative servers on multiple nodes in the domain.

Each resource in the WebSphere administrative domain corresponds to an object in the repository. For example, when you create an application, a corresponding application object is created in the repository. In this way, the administrative repository contents mirror the contents of the administrative domain.

The repository contains descriptive information about the resources that are configured to run on each node in the domain. For example, the repository contains the names of application servers, the node each server is running on, the enterprise beans installed in each server, and each server's current state (running, for example).

The repository allows you to administer the domain from any machine by storing the database in a central location. Each administrative server has a central view of configuration information about all resources in the domain. When you modify a resource's configuration, the changes are seen by all administrative servers.

The resources in a WebSphere administrative domain are represented in the administrative repository as entity beans with container-managed persistence (CMP). The persistent data associated with a resource (for example, the name, current state, and working directory of an application server) is stored in the administrative repository.

Administration occurs through method calls to resource beans in the administrative server. The WebSphere Administrative Console makes requests to an administrative server to access or modify a resource in the domain. An administrative server also communicates with other, remote administrative servers to delegate tasks and to respond to requests.

Session beans invoke methods on the resource beans. For example, you can start, stop, ping, and modify application servers in the WebSphere Administrative Console, which invokes methods on the resource beans for the

application servers. For more detailed information refer to the WebSphere Application Server Administrative Console online help.

9.2.2 Site Analyzer

The WebSphere Application Server Site Analyzer V3.0.2 provides analysis features and customizable reporting options that help you improve your Web site content and performance (content analysis), as well as better understand how a site is used by its visitors (usage analysis). Using Site Analyzer, you can quickly and easily report on everything from aggregate page sizes and broken links to site visit information and errors.

9.2.2.1 Site Analyzer configuration

Site Analyzer uses a client/server configuration. The server portion performs content and usage analysis and the client portion displays results of these analysis.

Note

If you plan to perform usage analysis, you need access to your HTTP server log files. In addition, you may need to reconfigure the HTTP server logs in order for Site Analyzer to process them correctly.

The client, server, and database portions can be installed on one machine (personal configuration) or on multiple machines (workgroup configuration). In the workgroup configuration, one system is the server, and one or more systems are the clients. For more detailed information about Site Analyzer refer to the WebSphere Application Server online help.

9.2.2.2 Starting Site Analyzer

If you are using the DB2 database provided with Site Analyzer, the first time the Site Analyzer server is started, you will be prompted to enter the following:

- database URL: jdbc:db2:sadata (local) or
dbc:db2://fully_qualified_host_name/sadata (remote)

If you are using a previously installed copy of IBM DB2 UDB (local or remote), you must configure DB2 with Site Analyzer before starting Site Analyzer.

Starting the Site Analyzer server

To start the server from the Windows NT GUI:

1. Click **Start -> Programs -> IBM WebSphere -> Site Analyzer 3.0**

2. Click **Yes** at the "Do you want to start the server?" prompt.

Note

If you only need to run the server, close the client and click **Yes** at the "Do you want to leave the server running?" prompt.

To start the server on AIX or Solaris, enter `wssas` at a command-line prompt. If you are using a DB2 database not installed by Site Analyzer, make sure you have modified the `wssas` script to refer to the correct DB2 instance.

The first time the Site Analyzer is started, a wizard for configuring Site Analyzer opens. To change the settings for the server at a later date, you can use the Preferences panel as follows:

- Windows NT:
Select **Options -> Preferences** in the Project Center
- AIX or Solaris:
The Preferences panel starts automatically when you start the server. (Enter `wssas` at a command line prompt.)

Tip

To avoid the configuration panel on AIX or Solaris, enter `wssas -noconfig` at a command-line prompt when starting the Site Analyzer server.

Starting the Site Analyzer client

To start the Site Analyzer client on Windows 95, 98, or NT:

- Click **Start -> Programs -> IBM WebSphere -> Site Analyzer 3.0**.

As with the Site Analyzer server, the first time you start the client a wizard will appear to configure general preferences.

9.2.2.3 Reporting

Using the Project Center

- Cut, copy, paste, and delete are not available for categories or aggregates.
- Some options are only available from the right-click menus.
- The Project Center window may appear when a secondary window is launched. If the window is present on the task bar, reselect it there. If not, select the **Project Center** from the task bar.

- Some time-consuming tasks do not always display an hourglass or other feedback that an operation has begun.

Creating reports

- To create charts in color in the final report, make sure the display is set to something other than *true color*; otherwise the charts will come out in black and white.
- Report element descriptions are limited to one line for every report element included in the project. Lines of text beyond this number might be truncated.
- When generating a report, the range values of Last 15 minutes, Last hour, and Last 24 hours only apply to report elements that pertain to measurements for the entire site. Data for all other report element types such as Resource or SubDomain are kept on a daily basis and is not detailed by hour or minute.
- If reports are regenerated from the sample project, a begin and end logo image must be specified for the report generator wizard.
- Use the Browse button to locate the sample begin and end logos in the samples directory.

AIX and Solaris only

- If you have installed other products that use the Install Toolkit for Java (ITJ), you may have problems uninstalling them after installing Site Analyzer V3.0.2.
- If you are running the Site Analyzer server, you must be a member of the groups db2iadm1 and db2asgrp. Otherwise, you will see an error message saying that you do not have enough privileges to perform requested actions.
- If you are an administrator and using a Windows client with an AIX or Solaris server, make sure you specify the same database user ID and password as you used to log in to the server.
- After installation you must edit wssas before starting Site Analyzer. Change the database instance name to invoke the correct database profile. See wssas located in the Site Analyzer main directory for details.

9.2.3 WebSphere Commerce Suite

The WebSphere Commerce Suite provides several utilities to help manage your commerce site.

9.2.3.1 Configuration Manager

The WebSphere Commerce Suite Configuration Manager tool has a Java-based graphical interface that lets you modify the way WebSphere Commerce Suite is configured without dealing with the intricacies of syntax-sensitive configuration files. Configuration Manager also makes it easy to control many of the administration tasks associated with WebSphere Commerce Suite.

Use this tool to perform the following tasks:

- Create a new WebSphere Commerce Suite instance
- Stop and start a WebSphere Commerce Suite instance
- Delete a WebSphere Commerce Suite instance
- Change the configuration settings for a WebSphere Commerce Suite instance
- Configure the database to serve as a staging server
- Enable or disable advanced caching
- Enable or disable basic caching

9.2.3.2 NCADMIN Management

The WebSphere Commerce Suite Administrator allows you to create and maintain virtual stores for selling products and services over the World Wide Web. It consists of the following components:

- Site Manager to manage mall level tasks.
- Store Manager to manage store level tasks.

Site Manager

The Commerce Suite Site Manager, referred to as the Site Manager, is a collection of online forms that you use to manage some high-level functions for e-commerce.

Use the Site Manager to establish a site, regardless of whether it is a single store or a mall. The person who manages these functions is called the site administrator. Use the Site Manager to do the following:

- Assign a mall header, footer, and home page.
- Assign site and store access by creating access control groups and assigning users to them to ensure that only authorized individuals are given access to the database.
- Maintain user IDs and passwords for site administrators and store administrators.

- Configure the site to support different currencies.
- Open or close a store.
- Define categories for the stores in the site, if the site has multiple stores.
- Assign scope to shopping tasks, such as shipping or ordering, to specify whether they will function similarly for all the stores in the mall, or vary from one store to the next.
- Assign Net.Data macros, JSPs or overridable functions to tasks to customize certain shopping processes.
- Define tax rates for stores.
- Manage information about shoppers.
- Maintain a list of shipping providers that are available to the stores.
- Assign security levels to commands.
- Configure personalization settings using the WebSphere Commerce Suite Administrator.
- Set up automated commerce reports.

Store Manager

The WebSphere Commerce Suite Store Manager, referred to as the Store Manager, is a collection of online forms that allow you to create and manage an online store. The person who manages these functions is called the store administrator.

Using the Store Manager, you can do the following:

- Enter information about the store.
- Assign headers and footers to store pages.
- Create product categories.
- Enter product information.
- Implement discounts.
- Select shipping services.
- Configure the payment module.
- Manage payment transactions.
- Create shopper groups.
- Create customized pages to display products and categories to shopper groups.
- Search for and view information about shoppers.

- Locate information about orders.
- Create auctions.
- Manage bids, manage auctions, and modify an auction.
- Create bid control rules and auction styles.
- Prepare an interactive catalog and shopping metaphors to help guide shoppers to product that best fits their needs.

Product Advisor

The Product Advisor provides an interactive environment for shoppers, by creating an *interactive catalog* that can contain the following elements:

- Marketing information, such as that possessed by a sales professional.
- Technical information, such as that possessed by a product specialist or subject matter expert.
- Shopping metaphors, to provide shoppers with different ways of finding what they want. Shoppers with little knowledge of a product category can use the sales assistance metaphor, which guides them toward appropriate products through a series of questions and answers. Those with more knowledge can use the product exploration metaphor, which lets them select desired product features from a list. Once the selection has been narrowed down through either of the above methods, shoppers can use the product comparison metaphor to compare similar products side by side.

The Product Advisor consists of the following components:

- The catalog builder, which prepares the product data for use in the electronic catalog.
- Three shopping metaphor builders, which let you set up different presentations of the catalog data.

Note

Product Advisor administration tools are designed to be used by a single user. There may be problems if the catalog builder or a metaphor builder is used by more than one person at once.

9.2.4 SecureWay Directory Management

The IBM SecureWay Directory Management Tool (DMT) provides a graphical user interface that enables you to manage information stored in directory servers to do the following:

1. Connect to one or more directory servers via SSL or non-SSL connections
2. Display server properties and rebind to the server
3. List, add, edit, and delete schema attributes and objectclasses
4. List, add, edit, and delete directory entries
5. Modify directory entry ACLs
6. Search the directory tree

For detailed information on SecureWay Directory integration with WebSphere Commerce Suite, refer to Chapter 14, “SecureWay Directory (LDAP) - AIX platform” on page 289.

9.2.5 HTTP Server Management

The HTTP Administration Server greatly simplifies the once-manual task of configuring your Web server. Once you select a server to configure, the Administration Server prompts you for configuration values, which are written to a configuration file when you click **Submit**.

9.2.6 DB2 UDB Management

With the DB2 administration tools, local or from remote database servers can be administered. These tools provide a graphical user interface for administration. The Control Center is the main DB2 graphical tool for administering your database. The Control Center provides a clear overview of all the systems and database objects being managed. You can also access other administration tools from the Control Center by selecting icons on the Control Center toolbar or from the tools pop-up menu.

9.2.6.1 Administration tools

The tools for administering DB2 are part of the DB2 administration client, a selectable component with each of the DB2 Universal Database products. The Administration Client is also available on a set of CD-ROMs that include the Administration Clients for all the operating systems on which DB2 is available.

They allow you to install and use the Administration Client on any workstation; it does not matter whether your database servers are local or remote, or what operating system the database servers are running on. The

tools enable you to perform the same functions from a graphical user interface as you could from the command line processor. These functions include the entering of DB2 commands, SQL statements, or system commands. With the tools, however, you do not have to remember complex statements or commands and you get additional assistance.

The following tools are available from the Control Center toolbar:

- Control Center

The Control Center is the main DB2 graphical tool for administering your database. From the Control Center, you get a clear overview of all the systems and database objects that are cataloged locally.

- Satellite Administration Center

The Satellite Administration Center allows you to administer DB2 Satellite servers.

- Command Center

The Command Center enables you to issue DB2 database commands, SQL statements, and operating system commands; recall previous commands; and scroll through access plans for SQL queries.

- Script Center

The Script Center enables you to create, run, and schedule operating-system-level commands and DB2 command scripts.

- Alert Center

The Alert Center notifies you when thresholds that you have set have been exceeded or when a node in a multinode environment is no longer responding.

- Journal

The Journal allows you to view the status of jobs and to view the recovery history log and messages log.

- Information Center

The Information Center gives you quick access to the information in the DB2 product manuals and sample programs and provides access to other sources of DB2 information on the Web.

- License Center

The License Center displays the status of your license as well as allowing you to configure your system for proper license monitoring.

DB2 provides SmartGuides for some of the common tasks performed using the GUI tools. SmartGuides are invoked from the pop-up menus in the Control Center. They provide a greater level of help by prompting you step-by-step on how to fill in the information necessary for the task you are doing and even making calculations and recommendations based on information you supply. SmartGuides are very useful if you are a new database administrator or someone who administers a database only occasionally.

In DB2 UDB, the following SmartGuides exist:

- Backup database

This SmartGuide asks you basic questions about the data in the database, the availability of the database, and recoverability requirements. It then suggests a backup plan, creates the job script, and schedules it. To invoke the Backup Database SmartGuide, select the icon representing the database you want to back up, click mouse button 2, and select **Backup -> Database using SmartGuide**.

- Create database

This SmartGuide allows you to create a database, assign storage, and select basic performance options. To invoke the Create Database SmartGuide, select the **Databases** icon in the Object Tree pane, click mouse button 2, and select **Create -> Database using SmartGuide**.

- Create table

This SmartGuide helps you to design columns using predefined column templates, create a primary key for the table, and assign the table to one or more table spaces. To invoke the SmartGuide, select the **Tables** icon, click mouse button 2, and select **Create -> Table using SmartGuide**.

- Create table space.

This SmartGuide lets you create a new table space and set basic storage performance options. To invoke it, select the **Table Space** icon, click mouse button 2, and select **Create -> Table space using SmartGuide**.

- Index SmartGuide

Use the Index SmartGuide to determine which indexes to create or drop for a given set of SQL statements. The recommendations are based on the workload that you specify. To invoke the Index SmartGuide, select the **Indexes** folder, click mouse button 2, and select **Create -> Index using SmartGuide**.

- Performance configuration

This SmartGuide helps you tune databases by requesting information about the database, its data, and the purpose of the system. It then recommends new configuration parameters for the database and instance and automatically applies them if you wish. To invoke this SmartGuide, select the icon for a database, click mouse button 2, and select **Configure using SmartGuide**.

- Restore database

This SmartGuide walks you through the process of recovering a database. To invoke the SmartGuide, select the icon for a database, click mouse button 2, and select **Restore -> Database using SmartGuide**.

- Configure multisite update

This SmartGuide lets you configure databases to enable applications to update multiple sites simultaneously when it is important that the data at all the sites must be consistent. To invoke this SmartGuide, select an instance, click mouse button 2, and select **Multisite Update -> Configure using SmartGuide**.

Note

SmartGuides do not exist for the DB2 for OS/390 subsystem.

Besides the graphical tools that you can invoke from the Control Center toolbar, there are some additional GUI tools that are not invoked directly from the Control Center toolbar.

- Performance Monitor. Performance Monitor is a tool to monitor DB2 objects such as instances, databases, tables, table spaces, and connections. You use this tool to detect performance problems and tune databases for optimum performance. The Performance Monitor is invoked as a selection on the pop-up menus in the Control Center.
- Event Monitor. Event monitor is a tool that lets you analyze resource usage by recording the state of the database at the time specific events occur. An Event Monitor is created by typing `db2emcrt` from a DB2 command line.
- Event Analyzer. Event Analyzer is a tool that allows you to analyze the data collected by the Event Monitor. An Event Analyzer is invoked by typing `db2evmon` from a DB2 command line.
- Visual Explain function. The Visual Explain function lets you view the access plan for SQL statements as a graph so that you can tune your SQL queries for better performance. Prior to Version 6, you used the Visual

Explain tool to view the access plans. In Version 6, Visual Explain is no longer a separate tool; however, the function is available on pop-up menus from various database objects in the Control Center, and also from the Command Center.

In addition to these tools, another useful tool for database administration that is not part of the Control Center is the Client Configuration Assistant. The Client Configuration Assistant is a SmartGuide with the primary function of setting up communications from remote clients to servers.

All these tools are described in greater detail in the online *DB2 Administration Guide*.

9.3 Security guidelines

Once your application is running in production mode, you can expect a lot of user access to the system. If we lived in a perfect world where all users were law abiding, then we would not have to worry about security. Unfortunately, this is not the case and your system will constantly face security threats, both external and internal.

The Web site you have developed is only one component of the total security system. The golden rule is that the security strength of your system is only as strong as its weakest link. Thus, it is necessary to ensure that the other components in the system are configured securely.

With this in mind, end-to-end security will consist of physical, operating system, network and application security as seen in Table 17 on page 207.

Table 17. End-to-end security components

Security type	Description
Physical	Control access to the hardware equipment hosting your application.
OS	Security at the operating system level.
Network	Secure connectivity flow among external, DMZ, and internal networks.
Application	Configure WebSphere security.

9.3.1 Physical systems security

Physical systems security is the foundation of the end-to-end security building blocks. Access to the hardware has to be controlled and monitored

proactively. Anyone gaining unauthorized physical access to your servers could halt your server, steal valuable information from your storage, plant viruses, install harmful software, etc. All of these activities are disruptive to your operations and will cause damage to your system. If the hardware is not secured properly, it will void the other security measures taken.

9.3.2 Operating systems security

After securing your physical systems, the operating system (OS) must be secured. As the OS grows richer in function and features, new bugs are discovered or are waiting to be exploited (for example, a bug that allows a user with non-privileged access to perform privileged operations).

At the operating system level, we recommend the following practices for administering your system:

1. Keep yourself updated on new information security glitches.

There are public Web sites that provide detailed information about newly discovered glitches. Fortunately, there is also a wealth of public information available which provides temporary or permanent remedies for these glitches. As an administrator, you need to be warned quickly about these loopholes and to take immediate actions to rectify the problem. As a service to their customers, most OS vendors provide updated information related to security hacks. A good source of information is the Web site by CERT (<http://www.cert.org>). You can subscribe to their mailing lists to receive regular updates and news flashes by e-mail.

2. Access privilege account needs

Your OS security policy will need to consider who has access to the privileged accounts and determine the roles and level of responsibility each person has. For example, you may want to separate the role of an OS administrator from the application administrator.

3. Enforce good password policies

Many security hacks are the result of simple passwords. You will have to enforce good password policies and practices for all accounts in your system, whether they are privileged or non-privileged. Besides having this policy, educate your users on their role in the overall system security. Another policy that you could implement, at the OS level, is forcing the passwords to expire or force them to change after some predefined period. Also you should not allow reuse of passwords.

4. Enable logging and auditing

Remember to turn on OS system logging and auditing. In the event of a system break-in, hopefully you will have some trails to start off your investigation.

9.3.3 Network security

Once you have the physical hardware and the operating system secured, you need to turn your attention to security between interconnected systems. Network security involves protecting resources residing in your internal network and DMZ from the external network. You need to restrict and prevent unauthorized user access to your internal systems. At the same time, you do not want to make it difficult for legitimate users to access your systems.

The key technologies available to achieve this network protection include firewalls, intrusion detection monitors, and anti-virus detection. The IBM SecureWay suite of products provide a comprehensive security solution that will meet most requirements. Information about SecureWay products can be found at <http://www.ibm.com/software/secureway/>.

9.3.3.1 WebSphere in a firewall environment

How do you restrict and control network access? You can use firewalls between two networks. When properly configured, the firewall acts as a choke point and will force all traffic of a specific protocol to and from the Internet to flow through it. By doing so, it can then scan the traffic and determine whether to allow or disallow the packets based on a set of rules.

When designing your firewall configuration we recommend the following:

1. Make sure that there is no direct communication channel between the applications on the intranet and the external Internet. In the example in Figure 39 on page 188, all external user requests and application responses flow through the Web server residing in the DMZ. If necessary, WebSphere Commerce Server will forward the request to the DB2 server in the internal network.
2. Keep it short and simple. Reuse pre-configured rules that exist. Define new rules if necessary. Always remember to include a rule that excludes everything else.
3. You should not allow information pertaining to the internal network to reach the Internet. For example, you would not want the IP addresses of your internal systems to be made available to external users. Hiding this information will reduce the risks of external security hacks. You can use network address translation to accomplish this.

4. At a minimum, you will need a firewall between the external network and your DMZ, and a firewall between the DMZ and the intranet. Introducing a DMZ configuration creates an additional security barrier which a network infiltrator would have to overcome.

Note

When you implement the DMZ configuration, it is possible to implement two or three firewalls on the same physical machine with two or three network adapter cards. This is only done for cost savings, but not preferred.

9.3.3.2 Internet and intranet security considerations

The intranet environment may be a LAN-based departmental network or could span geographic regions via a WAN-based Virtual Private Network (VPN). With this distinction in mind, we can see that the WAN-based intranet environment has similar characteristics to the Internet-based systems. The physical network used in the VPN network is usually outside your management control. Thus, you should continue to focus on network security.

For a LAN-based intranet that is segmented along departmental domains, you could still implement a firewall between the various departments. A good example would be to separate the production network environment from the development network environment.

9.3.4 Web application security

The user requests will flow through your firewall to your application. The final security checkpoint would be application security that decides who can invoke specific application function. WebSphere provides an integrated security model to configure security for your Web resources. This can be centrally configured through the WebSphere Application Server Administrative Console. Table 18 on page 210 lists the security components of the WebSphere Application Server.

Table 18. WebSphere security components

Security component	Location
Security plug-in	Supported Web server
Security collaborator	WebSphere Application Server
Security server	Security application

9.3.4.1 Security plug-in

When Web clients try to access Web resources deployed by the WebSphere Application Server, the security plug-in component will be invoked. The security plug-in will then send the client requests to the security collaborator for security decisions. This security plug-in is installed and configured with the Web server during software installation. Table 19 on page 211 lists the Web servers that WebSphere Application Server currently supports.

Table 19. Supported WebSphere Web servers

Web server	AIX	SOLARIS	Windows NT
Apache Server V1.3.6	X	X	X
Netscape Enterprise Server V3.51 and V3.61	X	X	X
Microsoft Information Server V4.0			X
Lotus Domino Application Server R5	X	X	X
Domino Go Webserver R4.6.2.5 and R4.6.2.6	X	X	X
IBM HTTP Server V1.3.6	X	X	X

9.3.4.2 Security collaborator

The security collaborator is attached to the WebSphere Application Server. It makes security decisions on remote method calls on servlets or EJBs by performing:

- Authorization checks
- Pre and post security trace logging
- Delegation policy enforcement

9.3.4.3 Security application

The security application resides in every WebSphere administrative server. It provides the means to configure security policies for both Web resources and EJBs. In a particular WebSphere Application Server Domain, there is a shared repository on the database server (DB2, Oracle) which contains all the security configuration and policy information. Any WebSphere Application Server in the same WebSphere domain can access this shared repository.

9.3.4.4 Security server

The security server is found in the security application. Both the security plug-in and security collaborator will call the security server for authentication/authorization services to provide:

- Centralized control over security policies (permissions, delegation)
- Central security services (authentication, authorization)

The security server is a trusted third party for security policy and control. Web servers and WebSphere Application Servers call on the security server to provide authentication, authorization and delegation services.

9.3.4.5 Secure the WebSphere Commerce Suite system

We recommend the following to secure the WebSphere Commerce Suite system:

1. Protect configuration files

Configuration files contain the host name, database name, password, and merchant key. If the Web server allows access to WebSphere Commerce Suite configuration files, they can be visible through Web browsers and become a security hazard. This information is so sensitive that WebSphere Commerce Suite provides a method to encrypt it.

2. Write secure Net.Data macros

When writing Net.Data macros follow the guidelines in *Net.Data Administration and Programming Guide for OS/2, Windows NT, and UNIX* found at <http://www.software.ibm.com/data/net.data/>.

Consider the following factors:

- Set configuration variables

Configuration variables limit the activities of end users and conceal the design of your database

- Define file access using path statements

Path statements identify one or more directories that Net.Data searches when attempting to locate macro files, executable files, include files, or other flat files. Define the directories on PATH statements to control the files that are accessible by browser users.

- Disable SHOWSQL

The SHOWSQL variable is used primarily for developing and testing the SQL within an application. It is not intended for use in a production system.

- Define request method.

The direct request method allows a user to execute a program directly within a URL, which may not be desirable. The macro request method

allows users to execute only those SQL statements and functions defined or called in a macro.

3. Disable samples and documentation

To increase the security of your production site, remove any databases used for testing or educational purposes.

4. Specify security level for commands

Security levels define whether SSL security and login authentication are required to run a particular WebSphere Commerce Suite command. The command security level can only be changed by the site administrator. Choose the appropriate option below:

- Assign security levels for commands
- Change the security assignment
- Remove the security assignment

9.3.5 WebSphere security model and policy

We have described the key infrastructure components in WebSphere security. Next, we would like to describe the WebSphere security model and policy. Understanding this allows you to make informed decisions in configuring and managing WebSphere security. For example, you can decide which of the authentication methods are supported given a particular user registry.

Specifically, we will describe the authentication, authorization, and delegation models and policies within WebSphere.

9.3.5.1 Authentication

Authentication is the process of proving a user is who they say they are. In WebSphere, authentication between a user and the WebSphere Application Server can be specified in terms of:

- User registry

This is where the user and group information will be stored.

- Authentication mechanism

After the user has provided the required data, the authentication mechanism will validate it against an associated user registry. Two types of authentication mechanisms are supported:

- Lightweight Third Party Authentication (LTPA)
- Native operating system

- Challenge Mechanisms

The challenge mechanism specifies how a server will challenge and retrieve authentication data from the user. It can be of the form:

- *None* - Security runtime does not challenge user for authentication data.
- *Basic* - A user is challenged for ID and password.
- *Certificate* - Mutual authentication over SSL.
- *Custom* - The ability to specify a custom HTML page to retrieve a user's ID and password.

Note

The components of the authentication are dependent on one another. For example, the authentication mechanism is defined based on the user registry and this choice drives the challenge mechanism.

In Table 20 on page 214 we illustrate the relationship between the user registry and the authentication mechanism. If user ID and password are supplied, then authentication is delegated to a user registry. If digital certificates are used, then the certificate credentials are mapped to an associated user registry entry.

Table 20. Mapping between authentication mechanism and user registry

Authentication	UNIX	Windows NT	LDAP
Native OS (user ID, password)	The supplied password is encrypted using the OS's crypt facility. This is then compared against the system's password repository.	Authentication is delegated to the Windows NT Security Access Manager via systems call.	N/A
LTPA (user ID, password)	N/A	N/A	An LDAP bind is performed using the DN (distinguished name) and the password.

Authentication	UNIX	Windows NT	LDAP
LTPA (digital certificate)	N/A	N/A	Based on the trust in the Web server, certificates are validated through successful establishment of a mutual SSL connection. A credential mapping is then performed based on the information contained in the certificate.

9.3.6 HTTP single sign-on (SSO)

When you enable HTTP single sign-on, user authentication credentials are preserved across multiple applications in the same domain, for example:

- Cooperating but disparate Web servers
- Cooperating applications such as the IBM OnDemand Server and Windows NT Suites.

With SSO, your application will then avoid repeated requests of user security credentials. However, if your application wants to use the SSO feature, then it must use an LTPA registry, (LDAP for example).

9.4 Backup and recovery guidelines

Backup may seem a mundane and repetitive task, but it is absolutely necessary. Its importance to you cannot be emphasized enough and typically you will only realize it during a disaster. Imagine losing valuable transactional data due to a hard disk failure and you do not have a backup.

We recommend the following when considering a backup solution:

1. Data to back up

You should consider the solution's support for the various data you need to back up. The data includes operating systems, application data, transaction logs, configuration files, application databases and WebSphere Application Server repository databases, and WebSphere Commerce Suite databases.

2. Available backup window time

In most situations, there is a limited window of time to complete the backup. Thus, you will have to consider the performance of the backup solution. Consider the performance of the solution as a whole, not the individual pieces.

3. Required system recovery time

Not only should the backup be fast, but the recovery process should be equally fast. Consider how the backup solution is able to provide fast recovery.

4. Support for enterprise backup

The solution should be scalable to perform backup of new systems that you may install, as a result of growth and upgrades. You may also want to use the backup solution for other existing applications.

5. Integration with system management tools

It is very useful if the backup solution can be integrated with existing system management tools and thus provides centralized administration.

6. Support for emerging technology

The software should be able to support emerging storage area network (SAN) solutions. As your informational needs grow, these storage solutions will provide large-capacity and high-performance data access.

9.4.1 Using Tivoli Storage Manager (TSM)

Based on the above factors in selecting a comprehensive backup solution, we recommend the IBM Tivoli Storage Manager (TSM). TSM is an integrated storage management solution that will meet the needs of any company, from small Internet startups to large enterprises. TSM provides the following features:

- Full support of client platforms
- High performance backup and recovery process
- Wide support for IBM and non-IBM tape/optical technology
- Scalable solution to meet growing storage demands
- Support for SAN-based solution
- Integrated with the Tivoli suite of systems management products

When using a Tivoli Storage Manager solution we recommend that each and every system in your configuration be backed up. The frequency of backup will vary, depending on the type of information and the frequency of changes.

Figure 40 shows the recommended Tivoli components for deployment in the internal network nodes.

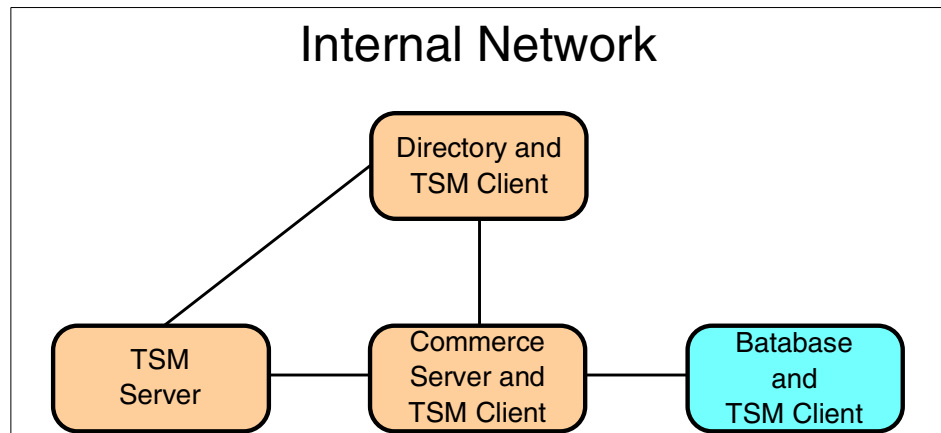


Figure 40. TSM server and client setup in the internal network

Referring to Figure 40, let's assume you install one TSM server in the scenario. This server should be located in the internal network with no external Internet access to it.

Once the TSM server setup is completed successfully, install TSM clients at every system that you would like to back up. For example, we have installed TSM clients at the directory server, commerce server, and the shared file system server. Test the connectivity between the TSM clients and the TSM server by doing a user-initiated backup.

For the servers in the DMZ as seen in Figure 41 on page 218, you can also install a TSM client. To facilitate communication between the TSM client in the DMZ and the TSM server in the intranet, you will have to open up one port in the firewall. This port can be preconfigured in both the TSM client and server.

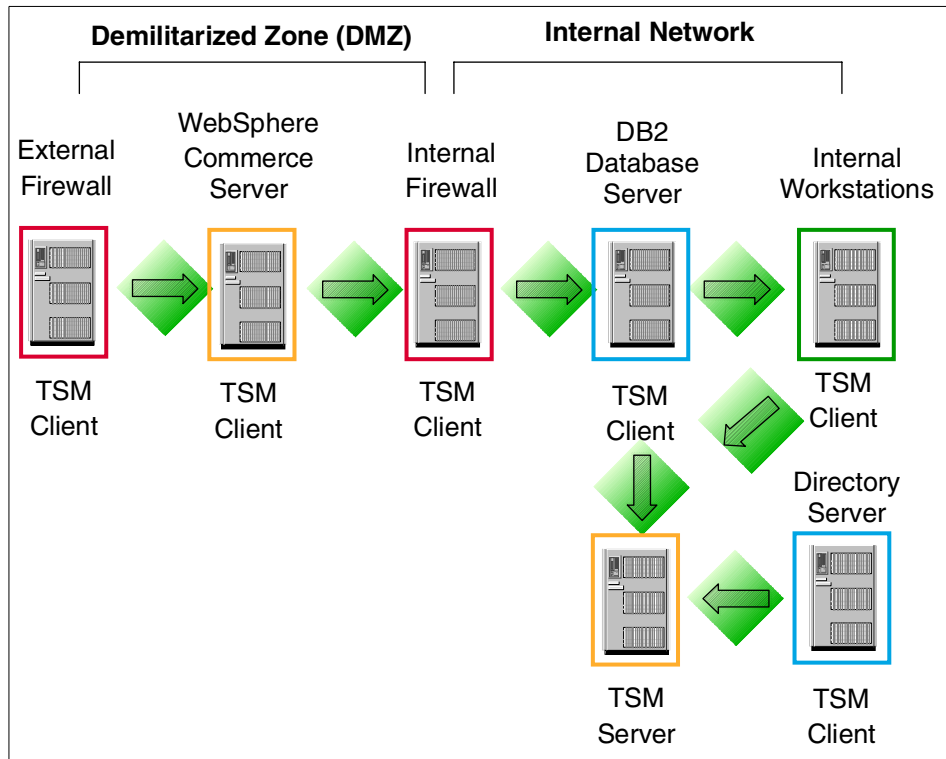


Figure 41. TSM server and client setup in the DMZ and internal network

This scenario is based on Figure 39 on page 188. If this additional firewall port is a security issue in your environment, then there are two viable options to consider:

1. Install Tivoli Data Protection for Workgroups (TDPfW) for Windows NT. TDPfW provides stand-alone disaster recovery for Windows NT machines. It can back up entire Windows NT machines or volumes, and if a disaster occurs, TDPfW can restore the complete machine, including the boot volume, disk partitions, security, operating systems, and user files from a locally attached SCSI tape drive.

TDPfW is very useful for small LAN environments. For example, systems in the DMZ shown in Figure 42, where you have to manage only a few servers, make good candidates for TDPfW. However, this product currently is supported only on Windows NT.

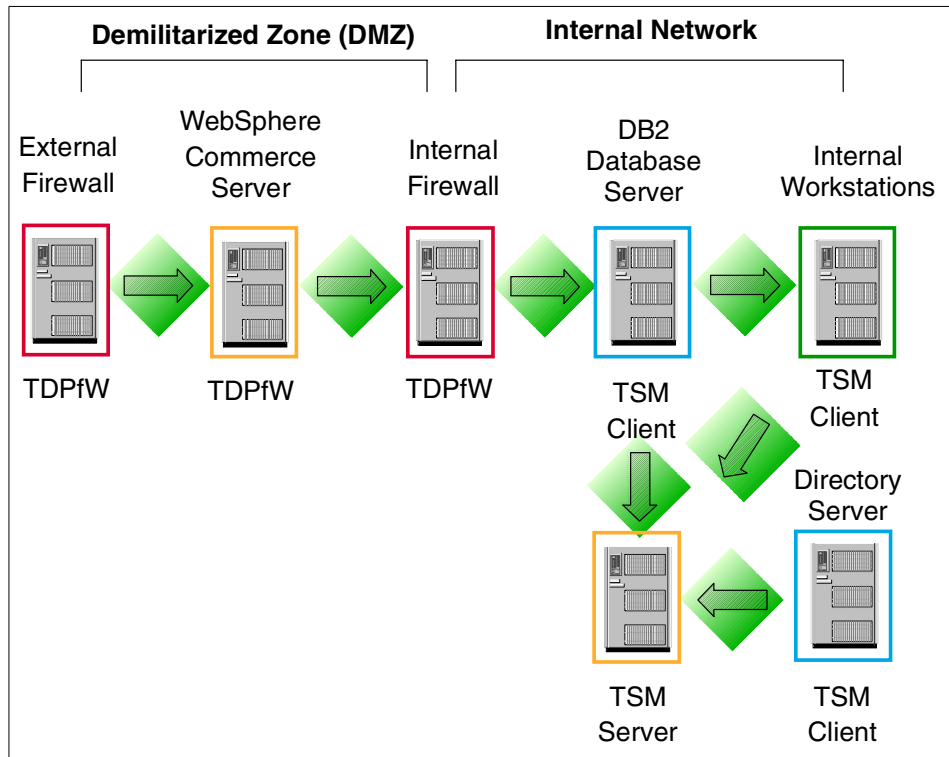


Figure 42. TSM server in the internal network with TDPfW in DMZ

2. Installing a TSM server in your DMZ

To facilitate a more automated solution, you could install another TSM server in the DMZ. Keep in mind that as more servers are added to the DMZ, the overhead involved in administering each TDPfW server increases.

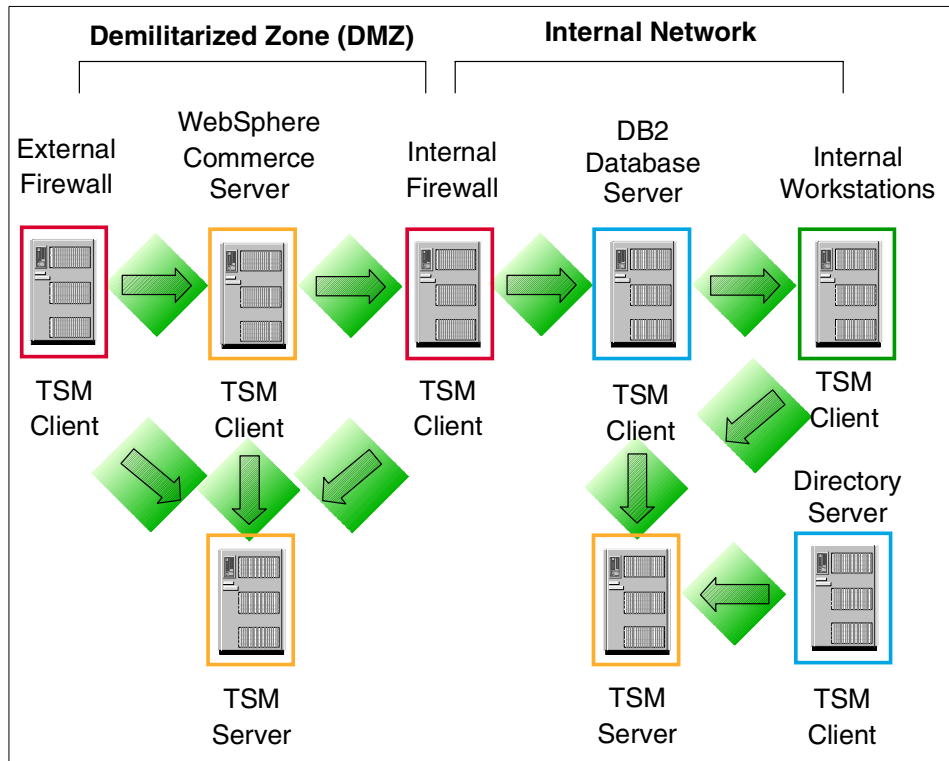


Figure 43. One TSM server in the DMZ and each internal network

9.4.2 Application backup and recovery

We have discussed general architecture for developing a backup solution. In this section, we will highlight product-specific areas to look out for.

9.4.2.1 Operating system

In a total system recovery scenario, the operating system (OS) will be the first software to be reinstalled. You should refer to your OS instruction manual for re-install steps. Our checklist below highlights practical tips for OS backup management:

- Do you have and know where the OS media are located?
- Do you have new updates to the OS?
- Do you have an OS backup after initial system setup?
- Do you do regular OS backup?
- Do you do ad-hoc OS backup before a major installation and after major configuration changes?

9.4.2.2 WebSphere database

When you install and configure the WebSphere Application Server, WebSphere Commerce Suite, SecureWay Directory, and WebSphere Site Analyzer, you will notice that they use DB2 as their database. In this section, we will briefly illustrate how a DB2 backup can be performed. For a comprehensive review of DB2 backup and recovery, please refer to the DB2 administrative guides and references.

In DB2, you can perform either an online or offline backup. When you do an offline backup, the database needs to be shut down. The command for an offline backup is:

```
db2> backup db <instance name> offline=yes
```

If shutting down the database is not an option, then you will have to perform an online database backup:

```
db2> backup db <instance name> online=yes
```

When performing an online database backup, you will have to take into consideration how the database logs will be managed, as they need to be used in a recovery process. Losing the logs will complicate the recovery process. When you use TSM, you will enjoy an automated database log backup solution. This can be achieved by:

- Configuring the DB2 user exit program to utilize TSM.
- Turning on the user exit and log retain parameters in DB2 database manager.

9.4.2.3 LDAP database backup

With the IBM SecureWay Directory, you can back up the LDAP database from the GUI interface, the command line, or via native DB2 commands.

Using the GUI as seen Figure 44 on page 222:

1. Click **Database** in the Navigation frame.
2. Click **Backup** in the Working with back-end databases in the work area.
3. Type the fully qualified name of the file to be created in the text entry field. This file will be in LDIF format.
4. Click the **Backup database** button to start backing up the database.

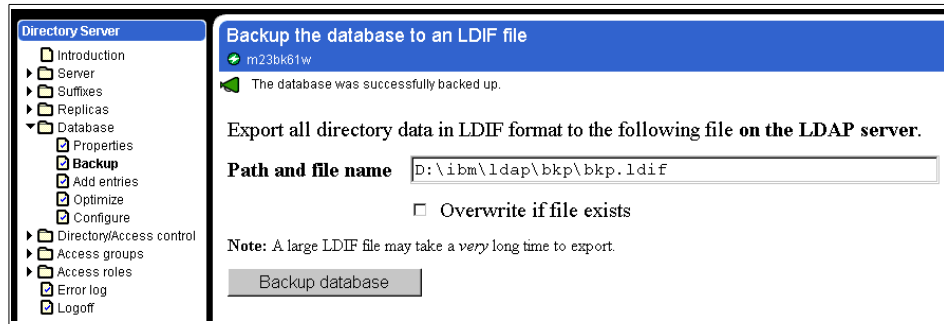


Figure 44. SecureWay Directory (LDAP) database backup GUI

Using the command line, the db2ldif tool can be used to dump entries from a directory to a text file in LDIF format. The syntax is:

```
C:\> db2ldif -o <output file> [-s <subtree>]
```

Note

LDIF format does not honor any ACL settings and all data is available in readable text.

9.4.3 Guidelines for backup and recovery

Independent of your backup software choice, we recommend the following:

1. Monitor the backup process

Ensure that the backup process is successful. If the backup process fails, understand why it fails and take necessary actions to remedy. For example, does the backup fail due to dirty drives or faulty media? Perhaps, there is a problem with the hardware? Could it also be a loss of connectivity between your backup server and the client machines?

2. Properly manage the backup media

After taking a backup of the system, ensure you have a proper and systematic tracking system for your backup media. In the event of a disaster, can you easily identify and retrieve these backup media for recovery?

3. Test your backup

Do not assume that if the backup process is successful, you have a valid backup. Test your backup! This will give you additional confidence that your backup can really help you recover from a disaster.

4. Document, test and maintain a recovery plan

No matter what size your system configuration may be, always have a documented recovery plan. This will prove useful during a disaster situation. The plan should describe step-by-step the process you will take to recover. Test the plan to ensure that it works! For the document to be useful at all times, you will have to maintain and update it when configuration changes occur.

9.5 Where to find more information

IBM Publications:

- *Application Server Solution Guide, Enterprise Edition: Getting Started*, SG24-5320
- *Patterns for e-business: User-to-Business Patterns for Topology 1 and 2 using WebSphere Advanced Edition*, SG24-5864
- *IBM WebSphere Commerce Suite Fundamentals*, GC09-2994
- *IBM WebSphere Commerce Suite Pro Edition, Commerce Suite Administrator V4.1*
- *Net.Data Administration and Programming Guide for OS/2, Windows NT, and UNIX* found at: <http://www.software.ibm.com/data/net.data/>

Other Publications

- General Web security information at: <http://www.cert.org/>.

Part 3. A working example

Chapter 10. Working example overview

The WebSphere Commerce Community library provides several complete working examples that can be found at:

<http://www.ibm.com/software/webservers/commerce/community/>. These working examples may be good if you do not have time to create everything from scratch.

In this working example we taken a different approach by providing instructions to create the store from scratch. We have designed the working example to provide step-by-step instructions for implementing the User-to-Online Buying Pattern application topology 2 (Enterprise-out) commerce site using WebSphere Commerce Suite V4.1. In the working example we use the fictitious *ITSO store* to implement the pattern.

Currently, the ITSO store sells books and CD's only in retail stores. The ITSO store needs the capability of selling its products online to retail stores at a discount, IBM employees at a discount, and to the general public at full price. The ITSO requires that the new online store have full integration to back-end systems for fulfillment, inventory, and customer order status.

The ITSO sample store is intended to be used for prototyping to help generate ideas for your business, and provide a base for creating a production level system. The ITSO online sample store uses many of the features of the WebSphere Commerce Suite. These features use various default commands and overridable functions provided in the WebSphere Commerce Suite to display store pages and perform specific business processes, such as adding items to the shopping cart and processing orders.

In the remaining chapters of this redbook we provide detailed instructions and discussion on the following topics:

- Runtime environment - AIX platform

Chapter 11, "Runtime environment - AIX platform" on page 229 will guide you through the installation, configuration and verification of a remote DB2 Database Server, and the WebSphere Commerce Server on the AIX platform. Appendix A, "Runtime environment - Windows NT platform" on page 423 provides the equivalent instructions for the Windows NT platform.

- Development environment

Chapter 12, "Development environment" on page 269 discusses the development tools, configuring the tools, and creating a development test environment.

- Create and publish a store
Chapter 13, “Create and publish a store” on page 273 provides step-by-step instructions for creating, and publishing the ITSO sample store.
- SecureWay Directory (LDAP) - AIX platform
Chapter 14, “SecureWay Directory (LDAP) - AIX platform” on page 289 discusses the integration of SecureWay Directory with the WebSphere Commerce Suite. In addition, we provide detailed instructions to guide you through the implementation of SecureWay Directory with WebSphere Commerce Suite on the AIX platform. Appendix B, “SecureWay Directory (LDAP) - Windows NT platform” on page 449 provides equivalent instructions for the Windows NT platform.
- Java technologies in WebSphere Commerce Suite
Chapter 15, “Java technologies in WebSphere Commerce Suite” on page 327 provides an overview of the Java technologies within WebSphere Commerce Suite. In addition, step-by-step instructions are provided to guide you through the creation and deployment of WebSphere Commerce JSPs, servlets, and beans.
- Personalization in WebSphere Commerce Suite
Chapter 16, “Personalization in WebSphere Commerce Suite” on page 379 provides instructions for personalizing your commerce site by using the rules engine with the WebSphere Commerce Suite.
- Back-end integration using MQSeries XML messages
Chapter 17, “Back-end integration using MQSeries XML messages” on page 401 provides instructions for implementing MQSeries XML message integration with WebSphere Commerce Suite for inbound and outbound messages.

Chapter 11. Runtime environment - AIX platform

This chapter provides installation and configuration instructions for the working example runtime environment on the AIX platform. We recommend using the working example runtime environment instructions in conjunction with the *IBM WebSphere Commerce Suite Pro Edition for AIX, Installation Guide V4.1* for specific questions about your runtime environment.

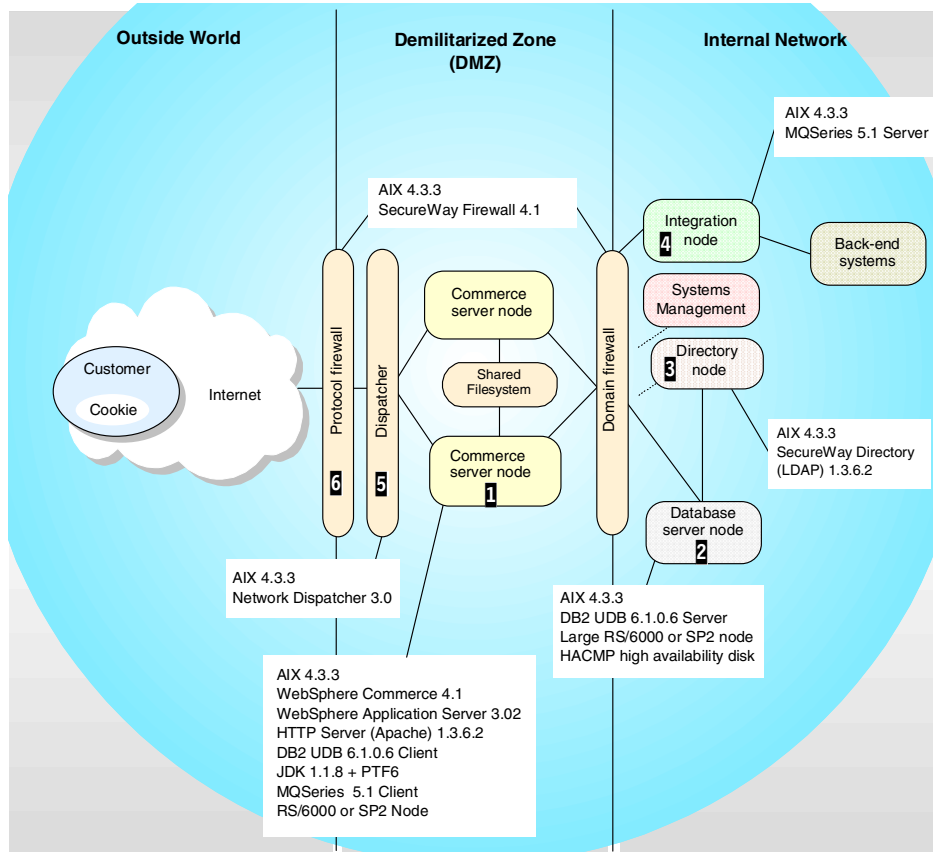


Figure 45. Working example runtime environment

Step-by-step instructions are provided for the following nodes as seen in Figure 45:

1. Commerce server node

Detailed instructions are provided for the installation, configuration, and verification of the commerce server node in 11.2, "WebSphere Commerce

server installation” on page 237, and 11.3, “WebSphere Commerce Suite configuration” on page 262.

2. Database server node

Detailed instructions are provided for the installation, configuration, and verification of the commerce server node and database server node in 11.1, “Remote DB2 database server installation” on page 230.

3. Directory node

Detailed instructions are provided for the installation, and configuration of the directory server node with the commerce server node in Chapter 14, “SecureWay Directory (LDAP) - AIX platform” on page 289.

4. Integration node

Detailed instructions are provided for the installation, and configuration of the integration node with the commerce server node in Chapter 17, “Back-end integration using MQSeries XML messages” on page 401.

5. Dispatcher node

Refer to *IBM WebSphere Performance Pack: Load Balancing with IBM SecureWay Network Dispatcher*, SG24-5858, and Chapter 16 in *Patterns for e-business: User-to-Business Patterns for Topology 1 and 2 using WebSphere Advanced Edition*, SG24-5864.

6. Firewall nodes

Refer to *A Secure Way to Protect Your Network: IBM SecureWay Firewall for AIX V4.1*, SG24-5855, and *WebSphere Scalability - WLM and Clustering*, SG24-6153.

This chapter is organized into the following sections:

- Remote DB2 database server installation
- WebSphere Commerce server installation
- WebSphere Commerce Suite configuration

11.1 Remote DB2 database server installation

The working example runtime environment provides detailed instructions for installing the remote DB2 database server for the AIX platform. The high level steps to setup the remote DB2 database server are as follows:

- Pre-installation requirements
- Pre-installation steps

- Install IBM DB2 UDB EE V6.1.0.6 for AIX

11.1.1 Pre-installation requirements

Prior to the installation of the remote DB2 database server, you must ensure that you meet the hardware and software requirements. Refer to chapter 1 of the *IBM DB2 Universal Database for UNIX, Quick Beginnings V6*, GC09-2836 for a detailed list of prerequisite hardware, software. In addition, refer to the *IBM WebSphere Commerce Suite Pro Edition for NT, Installation Guide V4.1*.

11.1.1.1 Hardware

In the working example, we used a RS/6000 F50 for the remote DB2 database server.

- IBM RS/6000 F50 (7025)
 - 333 MHz Power2 CPU, 4-way
 - 1 GB RAM
 - 9 GB DASD

11.1.1.2 Software - AIX operating system

In the working example, we installed IBM AIX V4.3.3.

11.1.2 Pre-installation steps

This section provides step-by-step instructions for preparing your system for a WebSphere Commerce DB2 server installation.

11.1.2.1 Verify paging space

You must have a minimum of 128 MB of paging space. By default, the AIX configuration assistant will suggest that you create a paging file twice the size of the RAM in your system. In the working example, we have 1 GB of RAM and a 2 GB paging file.

1. Log in to AIX as root, start a terminal session
2. Display paging space:

```
# lpsps -a
```
3. If the total active paging space is not greater than 128 MB, increase the paging space.

11.1.2.2 Increasing free space

The pre-installation requirements on page 7 of the *IBM WebSphere Commerce Suite Pro Edition for AIX, Installation Guide V4.1* assumes you are installing everything on one machine. When installing a remote database

server, the pre-installation requirements will be different. In the working example, only the DB2 server will be installed on the remote database server system.

1. Before increasing the file system size, view the existing free space by using the `df` command.

```
# df
```

2. Calculate the size required:

```
new_size = current_size + (desired_free_space - free_space)
```

3. Increase the size:

```
# chfs -a size='<new_size>' /<file_system_mount_point>
```

Table 21 provides a summary of the *IBM WebSphere Commerce Suite Pro Edition for AIX, Installation Guide V4.1* suggested values for blocks of free for each file system. In addition, the table provides the working example recommended blocks of free space.

Table 21. File system blocks of free space

File system	WCS V4.1 Install Guide - AIX suggested blocks of free space	Recommended blocks of free space
/ (root)	400000	100000
/home	400000	100000
/usr	2000000	1000000

Increase - / The / (root) directory may be used for temporary files during the installation. We increased / (root) directory by 100,000 blocks of free space.

Increase - /home By default, DB2 will create databases in the DB2 instance owner's home directory. In the working example we will create the WebSphere Commerce Suite and WebSphere Application Server Repository databases on a specified file system that we will allocate storage. For this reason we only increased /home by 100,000 blocks of free space.

Increase - /usr The /usr directory is the target directory of the DB2 application files. The DB2 fixed disk requirements for AIX can be found in *IBM DB2 Universal Database for UNIX, Quick Beginnings V6*, GC09-2836. In the working example, we installed the DB2 UDB Enterprise Edition with the default options, and increased /usr by 1,000,000 blocks of free space.

11.1.2.3 Network Information Services (NIS)

If you are using Network Information Services (NIS), you must disable it before installing DB2. If you do not disable NIS, the DB2 instance will not be created. You must complete this step in order to install DB2 successfully.

11.1.2.4 Create a logical volume and file system

The default database location for DB2 is the instance owner's home directory. In practice many users will create a file system to store their databases. Using a file system with access permissions provides the instance owner with control and the ability to set the storage size of the file system. In the working example we create a logical volume and file system to store the remote databases for the WebSphere Application Server and the WebSphere Commerce Server.

Create logical volume - "dblv1"

This procedure provides instructions for creating a logical volume on AIX.

1. Log in to AIX as root, start a terminal session
2. # smitty storage
3. Select **Logical Volume Manager** and press Enter
4. Select **Logical Volumes** and press Enter
5. Select **Add a Logical Volume** and press Enter
6. Select **VOLUME Group name:** and press F4 for a listing
 - Select **rootvg** and press Enter
7. When a Logical Volume window appears, enter the following fields:
 - Logical Volume NAME: **dblv1**
 - Number of LOGICAL PARTITIONS: **1**
 - Press Enter and wait until the status changes to Command: OK.
8. Press the F10 key to return to the system prompt

Create journal file system - mount point "/home/db2inst1"

Create a file system with the mount point to the home directory of the DB2 instance owner. The DB2 instance owner will be created in subsequent steps.

1. # smitty storage
2. Select **File Systems**
3. Select **Add / Change / Show / Delete File Systems**
4. Select **Journal File Systems**

5. Select **Add a Journaled File System on a Previously Defined Logical Volume**
6. Select **Add a Standard Journal File System**
7. When the Add a Standard Journal File System window appears, enter the following:
 - LOGICAL VOLUME name: press F4 for listing, select **dblv1**
 - MOUNT POINT: **/home/db2inst1**
 - Mount AUTOMATICALLY at system restart: choose **yes**

Tab to toggle

Press tab key to toggle yes/no.

- Press Enter and wait until the status changes to Command: OK.

8. Press the F10 key to return to the system prompt.

Allocate storage - “/home/db2inst1”

If you have not restarted your system, it is necessary to mount the file system prior to allocating storage to the newly created file system.

1. Mount file system:

```
# mount /dev/dblv1 /home/db2inst1
```

2. Allocate storage

This step will allocate file system storage space to the journal file system. By default, AIX allocates storage in 512 byte blocks. In this example we will create a file system that is 600,000 blocks or 300 MB. As the WebSphere Commerce Suite database, and the WebSphere Application Server repository database grow, this file system may need to be increased.

```
# chfs -a size='600000' /home/db2inst1
```

3. Verify file system free space:

```
# df
```

11.1.2.5 Create groups and users

Do the following to create groups and users:

1. Log in to AIX as root, start a terminal session
2. Create group db2iadm1 by typing the following:

```
# mkgroup db2iadm1
```
3. Create user db2inst1 by typing the following:

```
# mkuser pgrp=db2iadm1 db2inst1
```

4. Set default password for db2inst1 user

When creating a user, the password is set to * (for invalid). It is necessary to set the password. To change db2inst1 password type the following:

```
# passwd db2inst1
```

Changing password for "db2inst1"

```
db2inst1's New password: <your_password>
```

```
Enter the new password again: <your_password>
```

5. Change db2inst1 initial logon password

During the initial user logon, the user will be prompted to change a password. To change the initial logon password type the following:

```
# login db2inst1
```

```
db2inst1's Password: <your_password>
```

```
db2inst1's New password: <your_password>
```

```
Enter the new password again: <your_password>
```

6. Change ownership of /home/db2inst1 file system to user db2inst1 by typing the following:

```
# chown -fR db2inst1:db2iadm1 /home/db2inst1
```

7. Verify file system ownership (db2inst1:db2iadm1)

```
# ls -la /home/db2inst1
```

11.1.3 Install IBM DB2 UDB EE V6.1.0.6 for AIX

This section provides detailed instructions for installing the WebSphere Commerce remote DB2 database server.

1. Log in to AIX as root, start a terminal session

2. Mount the CD-ROM:

```
# mount -r -v cdrfs /dev/cd0 /mnt
```

3. Start DB2 install:

```
# cd /mnt
```

```
# ./db2setup
```

4. When the Install DB2 V6.1 window appears, select **DB2 UDB Enterprise Edition**, highlight **OK**, and press Enter.

5. When the Create DB2 Services window appears, select **Create a DB2 Instance**, highlight **OK** and press Enter.

6. DB2 instance authentication window appears, enter the following:
 - User Name: **db2inst1**
 - Group Name: **db2iadm1**
 - Home Directory: **/home/db2inst1**
 - Highlight **OK** and press Enter.

Note

The password field is intentionally left blank. In this example, we have manually created the user db2inst1 and set the password. DB2 can not change and existing users password. If a password is entered it will be ignored.

7. Message can not change password for an existing user, password will be ignored is displayed. Highlight **OK**, and press Enter.
8. Fence user window appears, enter the following:
 - User Name: **db2fenc1**
 - Group Name: **db2fadm1**
 - Home Directory: **/home/db2fenc1**
 - Password: **<your_password>**
 - Verify password: **<your_password>**
 - Highlight **OK**, and press Enter.
9. Create DB2 Services window appears, highlight **OK** and press Enter.
10. A Warning window displays The Administration server is not created. Highlight **OK** and press Enter.
11. The Summary Report window appears, listing the product components to be installed. Highlight **Continue** and press Enter.
12. Message This is your last chance to stop is displayed. Highlight **OK** and press Enter.
13. The db2setup program installs the selected components. Depending on the speed of your processor, this can take up to 15 minutes. When the install completes, a notice window informs you whether it was successful.
14. A notice window displays Completed successfully. Highlight **OK** and press Enter.
15. Scan the Status Report to ensure that all components were installed successfully. Highlight **OK** and press Enter.

16. The DB2 Installer window appears. Highlight **Close** and press Enter.
17. A Warning message The Administrative Server is not created is displayed. Highlight OK and press Enter.
18. The message Do you want to exit the DB2 Installer? appears. Highlight **OK** and press Enter.
19. The DB2 installation is now complete

11.1.3.1 Unmount the CD-ROM

To unmount the CD-ROM, do the following:

1. # cd /
2. # umount /mnt
3. Remove the DB2 UDB CD

11.2 WebSphere Commerce server installation

This section provides detailed instructions for installing the WebSphere Commerce server for the AIX platform. Detailed instructions for the Windows NT platform can be found in Appendix A, "Runtime environment - Windows NT platform" on page 423.

The WebSphere Commerce server installation includes the following server components:

- IBM AIX V4.3.3
- IBM JDK V1.1.8 + PTF6
- IBM HTTP Server (Apache) V1.3.6.2
- IBM DB2 UDB V6.1.0.6 Client
- IBM WebSphere Application Server Advanced Edition V3.02
- IBM Net.Data V6.1
- IBM WebSphere Commerce Suite V4.1

11.2.1 Pre-Installation requirements

Prior to the installation of the WebSphere Commerce server, you must ensure that you meet the hardware and software requirements. Refer to the *IBM WebSphere Commerce Suite Pro Edition for AIX, Installation Guide V4.1*, chapter 1 for details.

11.2.1.1 Hardware

In the working example, we used a RS/6000 43p for the WebSphere Commerce server.

- IBM RS/6000 43p (7043-150)
 - 375 MHz PowerPC CPU
 - 756 MB RAM
 - 9 GB DASD, 2

11.2.1.2 Software - AIX operating system

In the working example, we installed IBM AIX V4.3.3 with the default installation options.

Systems within the DMZ and secure networks need to have defined static routes. Add static routes on the AIX server and test connectivity to systems on the remote network before proceeding.

11.2.2 Pre-installation steps

11.2.2.1 Verify paging space

You must have a minimum of 128 MB of paging space. By default, the AIX configuration assistant will suggest that you create a paging file twice the size of the RAM in your system. In the working example, the WebSphere Commerce server had 512 MB of RAM and a 1 GB paging file.

1. Log in to AIX as root, start a terminal session
2. Display paging space:

```
# lpsps -a
```
3. If the total active paging space is not greater than 128 MB, increase the paging space.

11.2.2.2 Increasing free space

The pre-installation requirements on page 7 of the *IBM WebSphere Commerce Suite Pro Edition for AIX, Installation Guide V4.1* assumes you are installing everything on one machine. In the working example, the DB2 server is installed on a remote database server system and the WebSphere Commerce Suite is installed on a separate system.

1. Before increasing the file system size, view the existing free space by using the `df` command:

```
# df
```
2. Calculate the size required:

$new_size = current_size + (desired_free_space - free_space)$

3. Increase the size:

```
# chfs -a size='<new_size>' /<file_system_mount_point>
```

Table 22 provides a summary of the *IBM WebSphere Commerce Suite Pro Edition for AIX, Installation Guide V4.1* suggested values for blocks of free for each file system. In addition, the table provides the working example recommended blocks of free space on the commerce server.

Table 22. File system blocks of free space

File system	WCS V4.1 Install Guide - AIX suggested blocks of free space	Recommended blocks free space
/ (root)	400000	400000
/home	400000	100000
/usr	2000000	1500000

Increase - / The / (root) directory is used for temporary files during the installation. We increased / (root) directory by 400,000 blocks of free space on the WebSphere Commerce server.

Increase - /home In the working example databases will be created on the remote database sever on a specified file system. For this reason we recommend increasing /home only by 100,000 blocks of free space.

Increase - /usr We increased /usr by 15000000 blocks of free space. The DB2 server is installed on a remote database server system.

11.2.2.3 Network Information Services (NIS)

If you are using Network Information Services (NIS), you must disable it before installing DB2. If you do not disable NIS, the DB2 instance will not be created. You must complete this step in order to install IBM DB2 Universal Database 6.1.0.6 successfully.

11.2.2.4 Create a file system to store software downloads

The purpose of this procedure is to provide a storage location for software downloads such as Netscape, JDK, and PTFs.

Create Logical Volume - swlv1

1. Log in AIX as root and start an AIX terminal session

2. # smitty storage
3. Select **Logical Volume Manager**
4. Select **Logical Volumes**
5. Select **Add a Logical Volume**
6. Select **VOLUME Group name:** press F4 for a list
 - Select **rootvg**
7. When a Logical Volume window appears, enter the following fields:
 - Logical Volume NAME: **swlv1**
 - Number of LOGICAL PARTITIONS: **1**
8. Press Enter and wait until the status changes to Command: OK.
9. Press the F10 key to return to the system prompt.

Create Journal File System - mount point "/sw"

1. # smitty storage
2. Select **File Systems**
3. Select **Add / Change / Show / Delete File Systems**
4. Select **Journal File Systems**
5. Select **Add a Journaled File System on a Previously Defined Logical Volume**
6. Select **Add a Standard Journal File System**
7. When the Add a Standard Journal File System window appears, enter the following:
 - LOGICAL VOLUME name: press F4 for listing, select **swlv1**
 - MOUNT POINT: select **/sw**
 - Mount AUTOMATICALLY at system restart: select **yes**
8. Press Enter and wait until the status changes to Command: OK.
9. Press the F10 key to return to the system prompt

11.2.2.5 Allocate Storage - "/sw"

If you have not restarted your system, it is necessary to mount the file system prior to allocating storage to the file system.

1. # mount /dev/swlv1 /sw

This step will allocate file system storage space to the journal file system. By default, AIX allocates storage in 512 byte blocks. In this example we will create a file system that is 600,000 blocks or 300 MB.

```
2. # chfs -a size='600000' /sw
```

11.2.3 Install IBM JDK V1.1.8 for AIX

This section provides detailed instructions for installing the JDK on the WebSphere Commerce server system.

11.2.3.1 Selecting the JDK version

If you are using AIX 4.3.3, JRE 1.1.8 will be installed by default. The WebSphere Application Server ships with JDK 1.1.6. However, it supports JDK 1.1.8. We recommend that you use JDK 1.1.8 with PTF level 6 over JDK 1.1.6 for performance reasons.

If you decide to use JDK 1.1.6 supplied with the WebSphere Application Server CD, refer to the installation instructions in the *IBM WebSphere Commerce Suite Pro Edition for AIX, Installation Guide V4.1*.

11.2.3.2 Create a download directory for JDK 1.1.8 for AIX

On the WebSphere Commerce server do the following:

```
# cd /sw
# mkdir jdk118
# mkdir jdk118.ptf6
```

11.2.3.3 Download JDK 1.1.8

To use JDK 1.1.8 some additional steps must be followed. Download the JDK 1.1.8 packages for AIX found at <http://www.ibm.com/java/jdk/index.html> as follows:

1. From a browser enter URL: <http://www.ibm.com/java/jdk/aix/index.html/>.
2. Select **Register and Download**
3. Select version **1.1.8**.
4. Register or log in if you already have an ID
5. When a License window appears, click **I accept license**

- A JDK package list is displayed, select the tar file packages listed in Table 23, and download them to /sw/jdk118.

Table 23. JDK 1.1.8 for AIX downloads

Description	Tar filename
README (part of Java.rte)	JDK118.README.html
Core pieces (bin, classes, lib)	Java.rte.tar
Docs, includes, src for classes	Java.adt.tar
RMI-IIOP bin, docs, lib, samples	Java.rmi-iiop.tar
Demos, samples	Java.samples.tar (optional)
1.2 security backport docs, lib	Java.security.tar
Swing docs, examples, lib, src	Java.swing.tar

- Untar all the files:

```
# cd /sw/jdk118
# tar -xvf <filename.tar>
```

11.2.3.4 Download JDK 1.1.8 PTF 6

At the time of writing this book, JDK 1.1.8 PTF6 was the recommended service level. Download the JDK 1.1.8 PTF 6 packages for AIX from <http://service.software.ibm.com/rs6k/fixdb.html> as follows:

- From a browser enter URL: <http://service.software.ibm.com/rs6k/fixdb.html>.
- An AIX Fix Distribution window appears, enter JDK 1.1.8 in the search field and click **Find Fix**.
- When the Results from search window appears, select **JDK 1.1.8 PTF 6** and click **Get Fix Package**.
- When the Fix package from IBM window appears, select the entries in Table 24 and download the files to /sw/jdk118.ptf6.

Table 24. JDK 1.1.8 PTF 6 for AIX downloads

Description	Filesets
Docs, includes, src for classes	Java.adt.src.1.1.8.6
RMI-IIOP bin, docs	Java.rmi-iiop.docs.1.1.8.6
RMI-IIOP lib	Java.rmi-iiop.lib.1.1.8.6
Core pieces	Java.rte.bin.1.1.8.6

Description	Filesets
Core pieces	Java.rte.classes.1.1.8.6
Core pieces	Java.rte.lib.1.1.8.6
1.2 security backport docs, lib	Java.security.lib.1.1.8.6

11.2.3.5 Install JDK 1.1.8

To install JDK 1.1.8, do the following:

1. # cd /sw/jdk118.
2. # smitty install_all.
3. When the Install and Update from All Available Software window appears, enter the following in the Input device/directory for software field:
./
Press Enter.
4. In the Software to Install field, press F4 for a list. Highlight the following packages using F7 to select:
 - Java.adt
 - Java.rmi-iiop
 - Java.rte
 - Java.samples (optional)
 - Java.security
 - Java.swing
5. Press Enter.
6. When the Install and Update from All Available Software window appears, Press Enter.
7. When the Are You Sure message is displayed, press Enter to install.
8. Scan the log file to verify that the file sets were installed successfully. Press F10 to return to the system prompt.

11.2.3.6 Install JDK 1.1.8 PTF Level 6

To install JDK 1.1.8 PTF Level 6, do the following:

1. # cd /sw/jdk118.ptf6
2. # smitty install_all

3. When the Install and Update from All Available Software window appears, enter the following in the Input device/directory for software field:

```
./
```

 Press Enter
4. In the Software to Install field, press F4 for a list and highlight the following packages using F7 to select:
 - Java.adt
 - Java.rmi-iiop
 - Java.rte
 - Java.security
5. Press Enter
6. When the Install and Update from All Available Software window appears, press Enter
7. When the Are You Sure message is displayed, press Enter to install.
8. Scan the log file to verify that the file sets were installed successfully. Press F10 to return to the system prompt.

11.2.3.7 Verify the JDK 1.1.8 + PTF Level 6

To verify that the PTF level 6 is applied, type the following:

```
# java -fullversion
```

The following message will be displayed:

```
java full version "JDK 1.1.8 IBM build a118-20000210 (JIT enabled: jitc
V3.1-JDK1.1-20000210
```

11.2.4 Install IBM HTTP Server (Apache) for AIX V1.3.6.2

This section provides detailed instructions for installing the IBM HTTP Server on the WebSphere Commerce server system.

11.2.4.1 Installing IBM HTTP Server V1.3.6.2

1. Log in to AIX as user root and start an AIX terminal session.
2. Insert the WebSphere Application Server CD into the CD-ROM drive (contains the HTTP Server).
3. Mount the CD-ROM:


```
# mount -r -v cdrfs /dev/cd0 /mnt
```
4. # cd /mnt/usr/sys/inst.images

5. # smitty install_all
6. When the Install and Update from All Available Software window appears, enter the following in the Input device/directory for software field:
./
Press Enter
7. In the Software to Install field, press F4 for a list. Highlight the packages by pressing F7 to select. Refer to Table 25 for the list of packages to install.

Table 25. HTTP Server - Installable Packages

Package	Description
gskrf301 or gskru301	Global Security Kit (GSK) base function and 128-bit function respectively. If you select gskru301, it will prereq gskrf301 automatically.
http_server.admin	The server administrator used to configure the IBM HTTP Server.
http_server.base	The IBM HTTP Server.
http_server.frca	The Fast Response Cache Accelerator.
http_server.html. <i>locale</i>	The IBM HTTP Server documentation (where <i>locale</i> is your country code - for example en_US for U.S. English).
http_server.man.en_US	The IBM HTTP Server server manual. Note that this is available only in U.S. English.
http_server.modules Note: read the description for this package	The modules for IBM HTTP Server. Select the second and third modules ; if you simply select the entire module package your instance will not start. Module one is required only when using LDAP SSL integration with the HTTP Server.
http_server.msg. <i>locale</i> .admin	The message catalog (where <i>locale</i> is your country code). Choose en_US locale for U.S. English.
http_server.msg. <i>locale</i> .ssl.core	The message catalog, where <i>locale</i> is the code page value for your locale). Choose en_US locale for U.S. English.
http_server.ssl.128 or http_server.ssl.56	Determines whether you will have 128-bit encryption or 56-bit encryption.

Package	Description
http_server.ssl.core	Required in order to install SSL modules with encryption levels.

Note

Double check that you have select the correct packages. Due to the number of selections, it is easy to make a mistake which will result in not having your HTTP Server work properly.

8. Press Enter to close the list.
9. Press the Tab key to toggle to Yes to select the following:
 - Select **VERIFY install and check file sizes, Yes**
 - Select **DETAILED output, Yes**
10. Press Enter
11. When the Are You Sure message appears, press Enter.
12. The command status window appears. When the installation is complete, the Command field at the top of the window changes from Running to OK.
13. Scroll to the Installation Summary section at the bottom of the listing. In the Result column, you should see either SUCCESS or Already Installed next to the name of each component. If you do not, correct the problem.
14. Press F10 to return to the system prompt.

11.2.4.2 Unmount the CD-ROM

To unmount the CD-ROM, do the following:

1. # cd /
2. # umount /mnt
3. Remove the WebSphere Application Server CD

11.2.4.3 IBM HTTP Server Administrator - admin user/password

This HTTP Server Administrator user ID and password allow access to the Administration Server Configuration GUI.

1. # cd /usr/HTTPServer/bin
2. Create HTTP admin user (httpadm):


```
# ./htpasswd -m ../conf/admin.passwd httpadm
New password: <your_password>
Re-type new password: <your_password>
```

htpasswd syntax:

```
./htpasswd -m ../conf/admin.passwd <http_admin_user>
New password: <http_admin_user_password>
Re-type new password:<http_admin_user_password>
```

11.2.4.4 IBM HTTP Server - runtime user/password

The HTTP Server needs to be configured with a user to run after root has started the server. Run the setup script for IBM HTTP Server which creates a user ID and group, and changes the permissions for various configuration files. To set up IBM HTTP Server, perform the following steps:

1. # cd /usr/HTTPServer/bin
2. # ./setupadm
3. Answer the prompts as follows:
 - a. Please supply a user ID to run the Administration Server. For example:
 - User ID: httprun
 - Press Enter
 - b. Please Supply a Group Name to run the Administration Server. For example:
 - Group Name: httpgrp
 - Press Enter
 - c. Please supply the directory containing the files for which a change in the permissions is necessary. The default is /usr/HTTPServer/conf.
 - Press Enter to accept the default
 - d. Please supply the file name for permission changes. Default will change the file permissions for all files in the /usr/HTTPServer/conf directory.
 - Press Enter to accept the default
 - e. Special permissions are required for the Administration Server to handle the restart of managed Web servers. This will update the Group, Group access permissions, and Group execute permissions for the /usr/HTTPServer/bin/admrestart file. This interface is necessary for

the Administration server to manage Web servers. Enter 1 for YES (default) or enter 2 for NO.

- Press Enter to accept the default
- f. To perform the change, enter 1. To quit with no changes, enter 2 (default).
- Enter 1 to perform changes
 - Press Enter
- g. The configuration file `/usr/HTTPServer/conf/admin.conf` will be saved. Do you wish to update the Administration Server Configuration file? To perform the change, enter 1. To exit with no change, enter 2 (default).
- Enter 1 to update configuration file
 - Press Enter
- h. Do you wish to run the IBM HTTP Administration Server and IBM HTTP Server in a language other than English? For a language other than English, enter 1. For English, enter 2 (default).
- Press Enter to accept the default (English)
4. The `setupadm` program should return to a system prompt.

11.2.4.5 Enabling SSL for the HTTP Server

Enabling SSL is required for the WebSphere Commerce Suite.

Production SSL Enablement

Refer to Chapter 11 in the *IBM WebSphere Commerce Suite Pro Edition for AIX, Installation Guide V4.1* for instructions on production SSL enablement.

Test SSL Enablement

For the purposes of the working example, we will create a test security key file, as follows:

1. Stop the IBM HTTP Server:

```
# cd /usr/HTTPServer/bin
# ./apachectl stop
```
2. Copy sample `httpd.conf`

```
# cd /usr/HTTPServer/conf
# cp httpd.conf httpd.conf.orig
# cp httpd.conf.sample httpd.conf
```
3. Modify `httpd.conf` with a text editor

4. Uncomment the following lines by removing the # sign:

```
#LoadModule ibm_ssl_module libexec/mod_ibm_ssl_<encryption_level>.so
```

Note

Where the <encryption_level> is the appropriate level of your locale (for example, 128 for en_US).

```
#AddModule mod_ibm_ssl.c
#Listen 443
#<VirtualHost host.some_domain.com:443>
```

Note

You must substitute your fully qualified host name in this line (for example, <VirtualHost comaix.itso.ral.ibm.com:443>).

```
#SSLEnable
#</VirtualHost>
#SSLDisable
#Keyfile "/usr/HTTPServer/keys/keyfile.kdb"
```

Note

Replace the word **keys** with **ssl**.

```
#SSLV2Timeout 100
#SSLV3Timeout 1000
```

5. Save your changes.
6. Set Java path by typing the following:

```
# JAVA_HOME="/usr/jdk_base"
# export JAVA_HOME
```

7. Start IBM Key Management:

```
# ikeyman
```

8. When the IBM Key Management window appears, click the **Key Database File** menu and select **New**.

9. When the New window appears, enter the following:

- File name: keyfile.kdb

- File location: /usr/HTTPServer/ssl
 - Click **OK**
10. When the Password Prompt window appears, enter the following:
 - IBM HTTP Server runtime password: <your_password>
 - Select the **set expiration time** checkbox (change to desired number of days)
 - Select the **enable Stash the password to a file** checkbox
 - Click **OK**
 11. When the Information window appears, click **OK**.
 12. Click the **Create** menu and select **New Self-Signed Certificate**.
 13. When the Create New Self-Signed Certificate window appears, enter the following:
 - Key Label: ssl test certificate (user-defined name)
 - Organization: IBM (your company name)
 - Accept default for remaining fields, click **OK**
 14. Close the Key Management Utility.

Start HTTP Server

```
# cd /usr/HTTPServer/bin
# ./apachectl start
```

Start IBM HTTP Administration Server

```
# cd /usr/HTTPServer/bin
# ./adminctl start
```

11.2.4.6 Configure HTTP Server - ServerName

1. Start a Web browser; enter URL: http://<hostname>
2. When the Welcome to the IBM HTTP Server page is displayed, click **Configure server**.
3. IBM HTTP Administration Server login window is displayed, enter:
 - User ID: httpadm
 - Password: <your_password>
 - Click **OK** to continue.
4. Double-click **Basic Settings** from the left hand navigation bar.
5. Double-click **Core Settings**, enter the server hostname:

- Server Name: <fully_qualified_hostname>
 - Click **Submit** to save
6. Close the Web browser.

11.2.4.7 Verify HTTP Server

To verify the HTTP Server, complete the following steps:

1. Start a Web browser.
2. To verify the HTTP Server, enter URL: http:\\<your_hostname>
3. To verify that the HTTP Server is SSL enabled, enter URL:
https:\\<your_hostname>
4. Close the Web browser.

11.2.5 Install DB2 Client

This section provides detailed instructions for installing the DB2 Client on the WebSphere Commerce server system.

11.2.5.1 Installing the DB2 Client

1. Log in to AIX as user root and start an AIX terminal session.
2. Insert the DB2 UDB CD into the CD-ROM drive.
3. Mount the CD-ROM:

```
# mount -r -v cdrfs /dev/cd0 /mnt
# cd /mnt
```
4. Start DB2 install:

```
# ./db2setup
```
5. When the Install DB2 V6.1 window appears, select **DB2 Administration Client**, select **Customize**, highlight **OK**, and press Enter.
6. When the Create DB2 Services window appears, select **Create a DB2 Instance**, highlight **OK** and press Enter.
7. DB2 Instance window appears, enter the following:
 - User Name: db2inst1
 - Group Name: db2iadm1
 - Home Directory: /home/db2inst1
 - Password: <your_password>
 - Verify Password: <your_password>

- Highlight **OK** and press Enter
- 8. Create DB2 Services window appears. Highlight **OK** and press Enter.
- 9. When the Summary Report appears, listing the product components to be installed, press **Continue**.
- 10. A Warning message appears, This is your last chance to stop. Highlight **OK** and press Enter.
- 11. The db2setup program installs the specified components and creates the DB2 instance. Depending on the speed of your processor, this can take up to 15 minutes. When the install is complete, a notice window informs you whether the install was successful.
- 12. A notice appears, Completed successfully. Highlight **OK** and press Enter.
- 13. View the log to ensure that all components were installed successfully. Highlight **OK** and press Enter.
- 14. To close the DB2 Installer window, highlight **Close** and press Enter.
- 15. A notice appears, Do you want to exit DB2 Installer?. Highlight **OK** and press Enter to return to the system prompt.

11.2.5.2 Unmount the CD-ROM

To unmount the CD-ROM, do the following:

1. # cd /
2. # umount /mnt
3. Remove the DB2 UDB CD

11.2.6 Install WebSphere Application Server

This section provides detailed instructions for installing the WebSphere Application Server on the WebSphere Commerce server system.

11.2.6.1 Installation prerequisites

In order to continue with the installation, the following are required:

1. You must have the IBM HTTP Server, IBM DB2, and the supported JDK installed.
2. You must have a graphical terminal to install WebSphere Application Server.
3. You must stop the IBM HTTP Server and IBM HTTP Administration Server:

```
# cd /usr/HTTPServer/bin
```

```
# ./apachectl stop
# ./adminctl stop
```

11.2.6.2 DB2 server configuration

The following DB2 server configuration is required for the WebSphere Application Server repository database.

Create the WebSphere Application Server Repository database

On the remote database server, create the WebSphere Application Server (WAS) Repository database (wasaix) by typing the following:

1. Create the database by typing the following:

```
# su - db2inst1
$ db2 create db wasaix
```

2. Increase the appleheapsz for WAS

```
$ db2 update db cfg for wasaix using applheapsz 512
```

Note: applheapsz

Increase the applheapsz to 512 to improve performance

11.2.6.3 Commerce server - DB2 client configuration

This section provides instructions for configuring the DB2 client for connectivity to the DB2 server.

Catalog the TCP/IP node

On the WebSphere Commerce server type the following:

```
# su - db2inst1
$ db2 catalog tcpip node dbrvaix remote dbrvaix server 50000
```

Syntax

```
db2 catalog tcpip node <node_name> remote <db_server_hostname>
server 50000
```

Verify attach to the TCP/IP node

On the WebSphere Commerce server type the following:

```
$ db2 attach to dbrvaix user db2inst1 using <your_password>
```

Syntax

```
db2 attach to <node_name> user <db2inst_ID> using <db2inst_password>
```

Catalog the database

On the WebSphere Commerce server type the following:

```
$ db2 catalog db wasaix at node dbsrvaix
```

Syntax

```
db2 catalog db <database_name> at node <node_name>
```

Verify connect to database

On the WebSphere Commerce server type the following:

```
$ db2 connect to wasaix user db2inst1 using <your_password>
$ exit
```

Syntax

```
db2 connect to <db_name> user <db2inst_ID> using
<db2inst_password>
```

11.2.6.4 Install WebSphere Application Server Advanced Edition

On the WebSphere Commerce server, do the following:

1. Log in to AIX as root and start a terminal session.
2. Insert the WebSphere Application Server Advanced Edition V3.02 CD.
3. Mount the CD-ROM:

```
# mount -r -v cdrfs /dev/cd0 /mnt
# cd /mnt/aix
# ./install.sh
```
4. You will be prompted to enter your JAVA_HOME path; enter the following:

```
/usr/jdk_base
```
5. Welcome to the WebSphere Application Server Setup program window appears, click **Next**.
6. Prerequisite Check window appears, click **Next**.

7. Install Option window appears, select **Custom Installation**, and click **Next**.
8. Select Components and Subcomponents to Install window appears as seen in Figure 46 on page 255. Do the following to select the subcomponents:
 - Click the **Install and default configuration** checkbox
 - Click the **Production Application Server** checkbox
 With the Production Application Server subcomponent highlighted select the following:
 - Click the **Core Server** checkbox
 - Click the **IBM HTTP Server plug-in** checkbox
 - Click **Next**

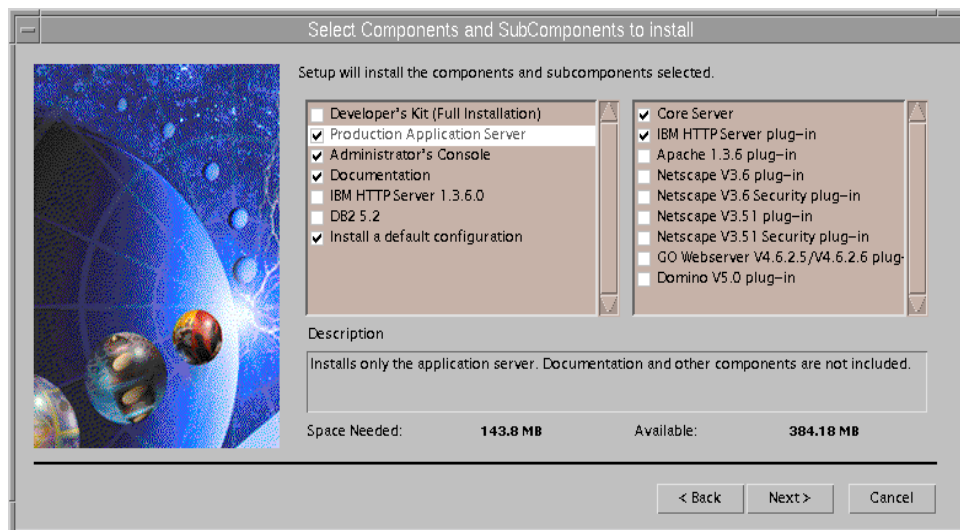


Figure 46. WebSphere Application Server Installation - Subcomponents

9. The User Information window appears. Enter information as seen in Figure 47 on page 256. When complete, click **Next**.

WebSphere User ID: wasadmin

WebSphere Password: <your_password>

Database Name: wasaix

Note

The database name must match the database created in 11.2.6.2, “DB2 server configuration” on page 253.

DB2 User ID: db2inst1 (instance owner)

DB2 Password: <your_password>

DB Home: /home/db2inst1 (home directory of instance owner)

DB URL: jdbc:db2:wasaix (this is generated based on the database name)

User Information

Please enter the data below and select Next.

Security

User ID: wasadmin Password: *****

Use Demo Key Ring

Client Key Ring: Password:

Server Key Ring: Password:

Database Information

Database Type: DB2 Database Name: wasaix

User ID: db2inst1 Password: *****

DB Home: /home/db2inst1 Browse

DB URL: jdbc:db2:wasaix

< Back Next >

Figure 47. WebSphere Application Server Installation - User Information window

10. When prompted to confirm that you want to begin copying files, click **OK** to begin the installation.
11. When the Setup Complete window appears, click **Finish**.
12. When the Readme File window appears, click **Finish**.

11.2.6.5 Unmount the CD-ROM

To unmount the CD-ROM, do the following:

1. # cd /
2. # umount /mnt
3. Remove the WebSphere Application Server CD.

11.2.6.6 Post-installation

After the WebSphere Application Server installation, the following steps must be completed.

Modify admin.config

If you have installed AIX V4.3.3, port 9000 will be in use by IBM X terminal in the services file. Since WebSphere Application Server attempts to use this port by default, you must change the port by performing the following steps:

1. # cd /usr/WebSphere/AppServer/bin
2. Modify admin.config with a text editor

Add the following line to the end of admin.config file:

```
com.ibm.ejs.sm.adminServer.lsdPort= 9001
```

Where 9001 (nnnn) is a port number not in use in the /etc/services file.

Start the WebSphere Application Server

1. To start the WebSphere Application Server, type the following:

```
# cd /usr/WebSphere/AppServer/bin
# ./startupServer.sh
```

2. WebSphere Application Server tracefile

Verify the WebSphere Application Server tracefile to see if the server started successfully by typing the following:

```
# cd /usr/WebSphere/AppServer/logs
# tail -f tracefile
```

If successful, you should see the following message in the tracefile:

```
031.257 600610b AdminServer A Initializing WebSphere Administration
server
031.279 602aa82 DrAdminServer A DrAdmin available on port 34,107
109.229 600610b InitialSetupI A Creating.Sample.Server.Configuration
156.067 600610b AdminServer A WebSphere Administration server open for
e-business
```

Configure default_host

To configure the default host, do the following:

1. Start the WebSphere Application Server Administrator Console:

```
# cd /usr/WebSphere/AppServer/bin  
# ./adminclient.sh &
```
2. When the Administrator Console window appears, select the **Topology** tab in the left-hand console frame.
3. Expand **WebSphere Administrative Domain** and select **default_host**.
4. Select the **Advanced** tab in the right-hand pane.
5. Under Host Alias, add fully qualified hostname of your WebSphere Commerce server. For example:
 - Host alias: comaix.itso.ral.ibm.com
 - Click **Apply**
6. For each entry under Host Alias you must enter the same entry with the secure server port number appended. For example:
 - Host alias: <hostname.some_domain.com>:443, click **Apply**
 - Host alias: <hostname>:443, click **Apply**
 - Host alias: <ip_address>:443, click **Apply**
7. From the left-hand frame, expand <your_hostname>
8. Under your hostname, select **Default Server** and click the green button on the console toolbar to start the Default Server.
9. When the Command “Default Server.start” completed successfully message appears, click **OK**.
10. Your WebSphere Application Server is now configured and started.

11.2.6.7 Start the HTTP Server

To start the HTTP Server, type:

```
# cd /usr/HTTPServer/bin  
# ./apachectl start
```

11.2.6.8 Verify WebSphere Application Server

To verify the WebSphere Application Server, from a Web browser, enter URL:
`https://<hostname>/servlet/snoop.`

You should see information about the following:

- Requested URL
- Init parameters
- Request information
- Request headers
- Client cookies
- ServletContext attributes

11.2.7 Installing IBM Net.Data for AIX V6.1

This section provides detailed instructions for installing IBM Net.Data for AIX V6.1 on the WebSphere Commerce server system.

1. Log in to AIX as root and start a terminal session.
2. Insert the IBM Net.Data 6.1 CD into the CD-ROM drive.
3. Mount the CD-ROM:


```
# mount -r -v cdrfs /dev/cd0 /mnt
# cd /mnt/code/aix
# smitty install_all
```
4. When the Install and Update from All Available Software window appears, enter the following in the Input device/directory for software field:


```
./
```

 Press Enter.
5. In the Software to Install field, press F4 for list, highlight the following packages using F7 to select:
 - Select **Net.Data**
 - Select **Net.Data.msg.locale** (where locale is your locale; for example the locale is en_US for U.S. English)
 - Press Enter to close the list
6. In the Detailed Output field, press Tab to toggle from no to yes, and press Enter.
7. To begin installation, press Enter.
8. Message Are You Sure? is displayed, press Enter.
9. When the installation is complete, the Command field at the top of the window changes from Running to OK.

10. After the installation has completed, scroll to the Installation Summary section at the bottom of the listing. In the Result column, you should see either SUCCESS or Already Installed next to the name of each component; if you do not, correct the problem.
11. Press F10 to return to the system prompt

11.2.7.1 Unmount the CD-ROM

To unmount the CD-ROM, do the following:

1. # cd /
2. # umount /mnt
3. Remove the IBM Net.Data V6.1 CD

11.2.8 Installing WebSphere Commerce Suite V4.1

This section provides detailed instructions for installing WebSphere Commerce Suite V4.1 on the WebSphere Commerce server system.

11.2.8.1 Installation prerequisites

1. Ensure that you have installed the following software on your WebSphere Commerce server system:
 - IBM JDK V1.1.8 + PTF6
 - IBM HTTP Server V1.3.6.2
 - IBM DB2 UDB V6.1.0.6 Client
 - IBM WebSphere Application Server Advanced Edition V3.02
 - IBM Net.Data V6.1
2. Ensure that the WebSphere Application Server is started prior to starting the WebSphere Commerce Suite V4.1 installation.

11.2.8.2 Install WebSphere Commerce Suite

On the WebSphere Commerce server complete the following steps:

1. Log in into AIX as root and start an AIX terminal session.

```
# mount -r -v cdrfs /dev/cd0 /mnt
# cd /mnt/CommerceSuite
# smitty install_all
```
2. When the Install and Update from All Available Software window appears, enter the following in the Input device/directory for software field:
./

Press Enter.

3. In the Software to Install field, press F4 for list and highlight the packages using F7 to select. For a list of all options to install, refer to Chapter 8 in the *IBM WebSphere Commerce Suite Pro Edition for AIX, Installation Guide V4.1*.

For the working example, we selected the packages listed in Table 26.

Table 26. Working Example WebSphere Commerce Suite Installation Packages

Package	Description
CommerceSuite.PRT	The Commerce Suite product registration
CommerceSuite.PZN	The Commerce Suite personalization
CommerceSuite.Server	The Commerce Suite server
CommerceSuite.loc. <i>locale</i>	Commerce Suite Administrator mall and sample mall, where <i>locale</i> is your language, such as en_US for U.S. English.
CommerceSuite.msg. <i>locale</i>	The message catalog, where locale is the code page value for your locale, such as en_US for U.S. English.

4. When you have made all your selections, press Enter.
5. In the Detailed Output field, press Tab to toggle from no to yes, press Enter.
6. When the Are you sure? message appears, press Enter.
7. When the installation is completed, scroll to the Installation section at the bottom of the listing. In the Result column, you should see either SUCCESS or "Already installed".
8. Press F10 to exit to return to system prompt

11.2.8.3 Unmount the CD-ROM

To unmount the CD-ROM, do the following:

1. # cd /
2. # umount /mnt
3. Remove the WebSphere Commerce Suite CD

11.3 WebSphere Commerce Suite configuration

This section provides detailed instructions for configuring the IBM WebSphere Commerce Suite, Pro Edition for AIX V4.1 with IBM DB2 UDB EE V6.1.0.6 for AIX.

11.3.1 Pre-configuration tasks

Complete the following steps before configuring the WebSphere Commerce Suite.

11.3.1.1 Copy netcpswd file

We need to copy the netcpswd file (decrypt tool) from the commerce server to the DB2 instance owner /sqllib/function directory on the remote database server.

On the DB2 server, complete the following steps:

1. Log in to AIX as root and start an AIX terminal session.

2. # su - db2inst1

3. \$ cd sqllib/function

4. \$ pwd

```
/home/db2inst1/sqllib/function
```

5. Copy the netcpswd file from the WebSphere Commerce server to the DB2 server instance owners sqllib/function directory. For example:

```
$ ftp <commerce_server_hostname>
```

```
User name: db2inst1
```

```
Password: <your_password>
```

```
ftp> cd /usr/lpp/CommerceSuite/nc_schema/db2/UDF/aix
```

```
ftp> bin
```

```
ftp> get netcpswd
```

```
ftp> bye
```

```
$ ls -l netcpswd
```

6. Change permissions of the netcpswd file

While logged in as the DB2 instance owner on the DB2 server system, change permissions on the netcpswd file, by typing the following:

```
$ chmod 755 netcpswd
```


11.3.1.2 Start WebSphere Application Server

On the WebSphere Commerce server complete the following steps:

1. Log in to AIX as root and start a terminal session.

2. Start the WebSphere Application Server:

```
# cd /usr/WebSphere/AppServer/bin
# ./startupServer.sh
```

3. Verify the WebSphere Application Server tracefile to see if the server started successfully by doing the following:

```
# cd /usr/WebSphere/AppServer/logs
# tail -f tracefile
```

4. Once the server is started, you should see the following message in the tracefile:

```
031.257 600610b AdminServer A Initializing WebSphere Administration server
031.279 602aa82 DrAdminServer A DrAdmin available on port 34,107
109.229 600610b InitialSetupI A Creating.Sample.Server.Configuration
156.067 600610b AdminServer A WebSphere Administration server open for
e-business
```

11.3.1.3 Start IBM HTTP Server

Ensure that the HTTP Server is started, if not complete the following steps on the WebSphere Commerce server system:

```
# cd /usr/HTTPServer/bin
# ./apachectl start
```

11.3.1.4 Start DB2 Server

Ensure that the DB2 server is started, if not complete the following steps on the DB2 server system:

1. Login as root and start an AIX terminal session.

2. If DB2 is not started, start by doing the following:

```
# su - db2inst1
$ db2start
$ exit
```

11.3.1.5 Install Netscape 4.61 browser

On a Windows client system, install the Netscape 4.61 Web browser provided on the WebSphere Commerce Suite CD. We recommend that you disable memory and disk caching in the browser preferences.

11.3.2 Create the WebSphere Commerce Suite Instance

There are three methods to create a WebSphere Commerce Instance.

1. Configuration Manager - Java application
 - Java application (start_config)
 - For more information refer to page 56 of the *IBM WebSphere Commerce Suite Pro Edition for AIX, Installation Guide V4.1*.
2. Configuration Manager - Java applet
 - Java applet (start_admin_server, browser)
 - For more information refer to page 57 of the *IBM WebSphere Commerce Suite Pro Edition for AIX, Installation Guide V4.1*

Note

We were unable to get this method working properly with a remote DB2 database and the WebSphere Application Server. When creating an instance using this method, the WebSphere Application Server repository database did not get updated with WebSphere Commerce entries.

3. Command line interface
 - For more information refer to page 71 of the *IBM WebSphere Commerce Suite Pro Edition for AIX, Installation Guide V4.1*.

Instance Creation

If your RS/6000 does not have a video adapter, using the Configuration Manager as a Java application is not an option. Also, you may not have physical access to the RS/6000, in which case the Java application would not work for you either.

For the working example runtime environment, we used the Configuration Manager - Java application (start_config).

11.3.2.1 Starting the Configuration Manager - Java application

Complete the following steps on the WebSphere Commerce server:

1. Start the Configuration Manager

```
# cd /usr/lpp/CommerceSuite/server/bin
# ./start_config
```

2. When the Configuration Manager logon window appears, enter the following:
 - User ID: webadmin (default)
 - Password: webibm (default)
 - You must change your password.
 - New password: <your_password>
 - Verify new password: <your_password>
3. When the Configuration Manager window appears, click **New** to create a new WebSphere Commerce Instance.
4. Select the **Database** tab of the Configuration Manager and enter the following information, as seen in Figure 48 on page 266:
 - Database Name: dbwcsaix
 - DBMS: IBM Universal Database
 - Instance Owner ID: db2inst1
 - Database User Logon: db2inst1
 - Database Logon Password: <your_password>
 - Database Option: (ignore this option for the working example)
 - Enable the **Use Remote Database** checkbox
 - Remote Database Location: dbsrvaix (remote database server hostname)
 - Remote Database Node Name: dbsrvaix

Use Configuration Manager to change the settings for WebSphere Commerce components.
You can accept the defaults or make changes to all enabled fields
Click the tabs to switch components.

Commerce Suite Server	Caching	Instance Data	Web Server
Payment	LDAP	Rule Server	Rule Service

Database Name:

DBMS:

Instance Owner ID:

Database User Logon:

Database Logon Password:

Database Option:

Use Staging Server

Use remote database

Remote Database location

Remote Database Node Name

Figure 48. Configuration Manager - database tab

5. Select the **Commerce Suite** tab of the Configuration Manager and enter the following information, as seen in Figure 49 on page 267:

- Instance Name: wcsaix
- Leave the remaining settings as default
- Click **OK** to create the WebSphere Commerce Suite instance

A successful WebSphere Commerce instance creation takes several minutes to create the instance, database, and populate the database with the WebSphere Commerce schema (tables, etc.). The following messages should be displayed in the Configuration Manager console:

```

WebSphere Commerce instance wcsaix has been created successfully.
Preparing database for WebSphere Commerce instance wcsaix.
Creating database dbwcsaix...
Database creation is completed. The log files are in
/usr/lpp/CommerceSuite/instance/wcsaix/logs
Starting the Commerce application server configuration.
The Commerce application server configuration was successful.

```

Use Configuration Manager to change the settings for WebSphere Commerce components.
You can accept the defaults or make changes to all enabled fields
Click the tabs to switch components.

Database	Payment	LDAP	Rule Server	Rule Service
Commerce Suite Server	Caching	Instance Data	Web Server	

Instance Name:

Communication Port Base:

Number of Server Processes:

Authentication Mode: X.509
 Basic

WebSphere Data Source:

Encryption: Use Credit Card Encryption
 Use Default Merchant Key

Merchant Key:

OK Cancel Help

Figure 49. Configuration Manager - commerce suite tab

6. Click **OK**, select instance, and click **Start** to start the WebSphere Commerce Suite instance.
7. Click **OK**, The Server Status should say **Active** in green text (inactive in red).

11.3.3 Post-configuration steps

11.3.3.1 Start WebSphere Application Server

If not already started, start the WebSphere Application Server by typing the following:

```
# cd /usr/WebSphere/AppServer/bin
# ./startupServer.sh &
```

11.3.3.2 Stop/start IBM HTTP Server

To stop and start the HTTP Server, complete the following steps:

```
# cd /usr/HTTPServer/bin
# ./apachectl stop
# ./apachectl start
```

11.3.3.3 Start IBM HTTP Administration Server

To start the HTTP Administration Server, type the following:

```
# cd /usr/HTTPServer/bin
# ./adminctl start
```

11.3.3.4 Start WebSphere Commerce server

On the WebSphere Commerce server complete the following steps:

1. Start the WebSphere Application Server Administrative Console by typing the following:

```
# cd /usr/WebSphere/AppServer/bin
# ./adminclient.sh &
```

2. Select the **Topology** tab in the left-hand console frame.
3. Expand **WebSphere Administrative Domain**. Select and expand `<your_hostname>`.
4. Select **WebSphere Commerce Server** on the left-hand console frame under your hostname.
5. Click the green button on the console toolbar to start the WebSphere Commerce server (application server).
6. When the Information dialog window displays Command “WebSphere Commerce Server.start” completed successfully, click **OK**.
7. On the right-hand frame, the Current State should say Running.

11.3.3.5 Verify WebSphere Commerce Suite

The following steps are provided to verify that the WebSphere Commerce Suite is working properly:

1. On a Windows client, start the Netscape 4.61 browser. Enter URL:
`http://<your_hostname>/ncadmin`
2. When the WebSphere Commerce Suite - ncadmin window appears, enter the following:
 - User Name: ncadmin (default user)
 - Password: ncadmin (default password)

For security reasons, please change your password to something other than ncadmin. Enter your new password.
3. The WebSphere Commerce Suite Administration window should be displayed. We will explain how to use this tool in greater detail in later sections of the working example. Continue to the next chapter.

Chapter 12. Development environment

This chapter is organized into the following sections:

- WebSphere Commerce Suite test environment
- WebSphere Commerce Studio
- C++ CGI development

12.1 WebSphere Commerce Suite test environment

For the purposes of unit testing, we recommend the following for installing and configuring your WebSphere Commerce Suite test environment.

1. 1-tier - standalone WebSphere Commerce Suite server:
 - Windows NT platform
Refer to the *IBM WebSphere Commerce Suite Pro Edition for NT, Installation Guide V4.1*, packaged with the WebSphere Commerce Suite V4.1 product.
 - AIX platform
Refer to *IBM WebSphere Commerce Suite Pro Edition for NT, Installation Guide V4.1*, packaged with the WebSphere Commerce Suite V4.1 product.
2. 2-tier and 3-tier - separate commerce server and database server
 - AIX platform
Refer to Chapter 11, “Runtime environment - AIX platform” on page 229 in conjunction with *IBM WebSphere Commerce Suite Pro Edition for AIX, Installation Guide V4.1*, packaged with the WebSphere Commerce Suite V4.1 product.
 - Windows NT platform
Refer to Appendix A, “Runtime environment - Windows NT platform” on page 423 in conjunction with *IBM WebSphere Commerce Suite Pro Edition for NT, Installation Guide V4.1*, packaged with the WebSphere Commerce Suite V4.1 product.

12.2 WebSphere Commerce Studio

This section provides an overview of the WebSphere Commerce Studio and detailed instructions for installation.

12.2.1 WebSphere Commerce Studio overview

The WebSphere Commerce Studio provides a wide array of development tools. There are some highly sophisticated designer environments and some wizard-driven simple tools. Figure 50 provides a summary of applications and modules that come with WebSphere Commerce Studio. These tools can be used to develop the entire commerce store from scratch or modify an existing store.

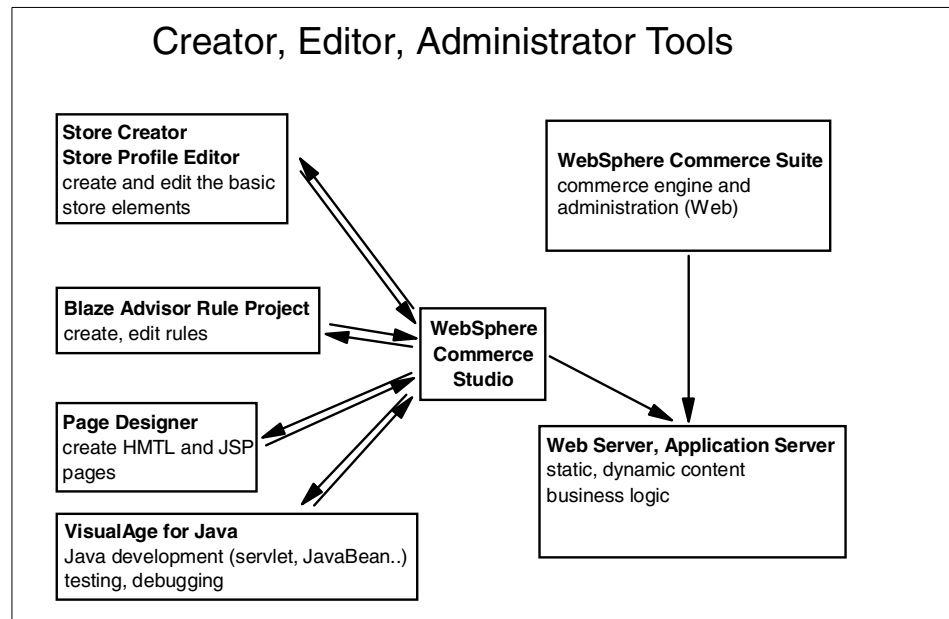


Figure 50. Tools in WebSphere Commerce Suite

The WebSphere Commerce Studio, Professional Developer Edition includes the following tools:

- WebSphere Studio
- Store Creator
- Store Profile Editor
- Page Designer
- Blaze Advisor rule projects
- Import wizard
- SQL wizard, Database wizard, JavaBean wizard
- JAR wizard

- Animated GIF Designer, WebArt Designer
- VisualAge for Java
- HotMedia
- Perfect Photo
- WebSphere Commerce Suite

For more detailed information refer to 8.9.2, “Development tools” on page 170.

12.2.2 WebSphere Commerce Studio installation

1. Insert the WebSphere Commerce Studio, Pro Edition CD into the CD-ROM drive of your Windows NT development system.
2. From the root directory of the WebSphere Commerce Suite, Pro Edition CD, run setup.exe.
3. Welcome window appears, click **Next**.
4. Software License Agreement window appears, click **Accept** to accept the terms of the agreement.
5. Installation Options window appears, select the following checkboxes:
 - **Applet Designer**
 - **VisualAge for Java**
 - **IBM Distributed Debugger**
 - Click **Next** to continue
6. Choose Destination window appears, accept the defaults and click **Next**.

Destination directory

Reducing the number of characters used in the directory path name will allow for faster file system related tasks.

7. Select Program Folder window appears, accept default and click **Next**.
8. Summary window appears, listing the components to be installed, click **Next**.

Product Destinations

Take note of the directories and products that will be installed.

9. If your browser configuration does not include localhost:49213 and 127.0.0.1 a window will appear asking you to update your proxy settings. Click **Yes** to continue.
10. Setup Complete window appears, select **Yes, I want to restart my computer now**. Click **Finish**.
11. Installation is now complete.

12.2.3 VisualAge for Java configuration

To take full advantage of the features provided by VisualAge for Java some additional configuration is necessary for the following:

- Adding features
- WebSphere Test Environment (WTE)
- JSP Execution Monitor

For detailed instructions on how to configure and use these features, refer to *Servlet and JSP Programming with IBM WebSphere Studio and VisualAge for Java*, SG24-5755.

12.3 C++ CGI development

Overridable functions (OFs) in WebSphere Commerce Suite are written in C++ and require a platform-specific compiler. These development tools are not shipped with WebSphere Commerce Studio. Table 27 provides a list of C++ compilers required.

Table 27. C++ compilers

Operating System	C++ compiler
IBM AIX 4.3	IBM VisualAge C++ 3.6.4
Microsoft Windows NT 4.0	Microsoft Visual C++ 6.0

For information on C++ compilers refer to <http://www.ibm.com/software/webservers/commerce/servers/versions.html/>.

Chapter 13. Create and publish a store

There are several ways to create a WebSphere Commerce Suite store. For the working example, we used the Store Creator as a starting point to create the base function commerce store.

This chapter is organized into the following sections:

- Creating a store using Store Creator
- Publishing a WebSphere Commerce Studio project
- Customizing a store

13.1 Create a store using Store Creator

In this section, we will create the ITSO sample store by using Store Creator.

1. Start WebSphere Commerce Studio:
 - Click **Start** -> **Programs** -> **IBM WebSphere Commerce Studio** -> **Studio 3.0** -> **IBM WebSphere Studio V3.0.2**.

Note

The first time you start WebSphere Studio, the Welcome to IBM WebSphere Studio window will appear. Then proceed to step 2.

2. Create a new project
 - Select **Create a new project**, click **OK**. If you have already created a project, select **File** -> **New Project**.
3. When the New Project window appears, enter the following:
 - Project Name: **itso**
 - Project Folder: **C:\IBM\WebSphereStudio\Projects\itso**
 - Project Template: **Store**
 - Click **OK**
4. When the WebSphere Commerce Studio - Store Creator window appears, click **Next**. The Store Creator walks you through the process of creating a base function commerce store.
5. When the Select a store model window appears, do the following:
 - Select Store model: **Retail Store**
 - Click **Next**.

6. When the Enter store information window appears, enter the following:
 - Store name: **ITSO**
 - Store directory: **itso**
 - Default currency: **US Dollar (USD)**
 - Click **Next**.
7. When the Enter contact information window appears, enter contact info, and then click **Next**.
8. When the Select a registration type window appears, accept the defaults, and then click **Next**.
9. When the Select payment option window appears, select the following:
 - Select **Accept orders without payment information (offline payment)**
 - Select **Accept orders with payment information (online payment)**
 - Select the credit cards that your store accepts:
 - **American Express, Master Card, Visa**

Credit Card Selection

To select more than one credit card from the list, press the Ctrl key, then select the credit card from the list with the mouse pointer.

- Deselect **Shopper completes payment page and pays with an online wallet using SET**
 - Select **Shopper completes payment page only**
 - Click **Next**
10. When the Select a store style window appears, accept the defaults, and then click **Next**.
 11. When the Select location of navigation bar window appears, select **Side (left)**, and then click **Next**.
 12. When the Select images window appears, accept the defaults, and then click **Next**.
 13. When the Select colors and background images window appears, accept the default, and then click **Next**.
 14. When the Sample products, taxes and shipping window appears, do the following:
 - Select **Add sample products to store**

- Select **Add sample taxes to store**
- Select **Add sample shipping to store**
- Click **Finish**

15. When the Where to go from here window appears, read the information displayed, and then click **Close**.

Attention

Do not publish the project at this time. The publishing paths will be modified in subsequent steps.

16. Insert image files to the images folder in studio. These images will be used as icons for products, categories, products, nav bars, etc..

13.2 Publishing a WebSphere Commerce Studio project

Once your base commerce store has been created, you need to publish the contents of the site to the target host system. This section provides detailed instructions for the following tasks:

- Defining the target publishing host
- Configuring the target publishing host - AIX platform
- Configuring the target publishing host - Windows NT platform
- Configuring the WebSphere Commerce Server
- Verifying the base store

13.2.1 Defining the target publishing host

Depending on the target host system type, Windows NT or AIX, there are different methods required for configuring your target publishing host. We have provided detailed instructions for publishing a project to an AIX host. Many of the steps apply to both platforms.

To define the target publishing host, complete the following steps:

- Add a server
- Set the default server
- Modify the store
- Configure the WebSphere Application Server

13.2.1.1 Add a server

From the WebSphere Commerce Studio store, complete the following steps:

1. Select **View -> Publishing**.
2. Select **Project -> Publishing Stage -> Test**.
3. Select **Test** from the right-hand frame, and from the menu bar select **Insert -> Server**.
4. When the Insert Server window appears, enter the following:
 - Server Name: <your_full_qualified_hostname>

13.2.1.2 Set default server

1. Select **Test** from the right-hand frame, right-click and select **Properties -> Advanced** tab.
2. From the Advanced tab, do the following:
 - Default server: select <your_hostname>
 - Click **OK**

13.2.1.3 Modify the store

1. From the left-hand frame, select and expand **stores**.
2. Select the **theme** folder and drag and drop the **theme** folder within <your_store> folder.
3. Create a jsp folder within <your_store> folder:
 - a. From the left-hand frame, select <your_store>, right-click -> **Insert -> Folder**.
 - b. When the Insert Folder window appears, enter the folder name jsp and then click **OK**.
4. Create your store folder within the servlets folder:
 - a. From the left-hand frame, select the **servlet** folder, right-click -> **Insert -> Folder**.
 - b. When the Insert Folder window appears, enter the folder name itsa (your store), and then click **OK**.
 - c. Select <your_store> folder under **servlet**, right-click -> **Insert -> Folder**.
 - d. When the Insert Folder window appears, enter the folder name servlets, and then click **OK**.
5. Your WebSphere Commerce Studio Project should look like Figure 51 on page 277.

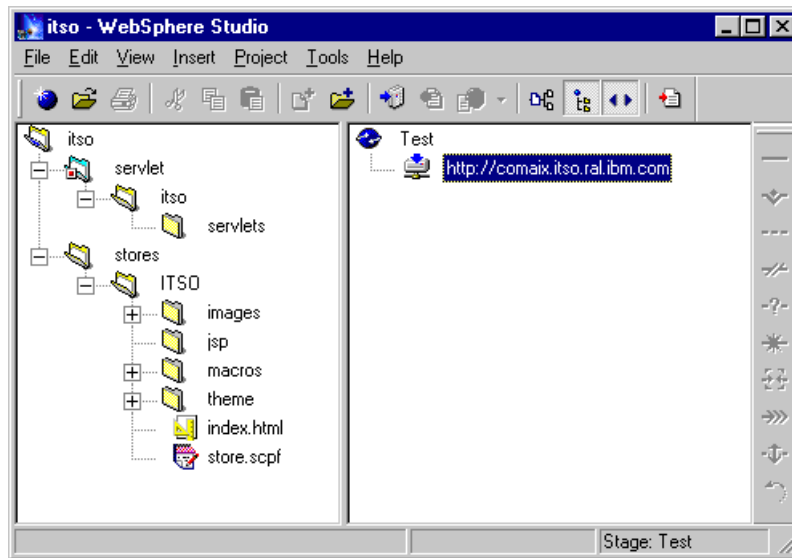


Figure 51. WebSphere Commerce Studio store modifications

13.2.2 Configuring the target publishing host - AIX platform

This section provides detailed instructions for configuring the target publishing host for an AIX WebSphere Commerce server. The configuration includes the following steps:

- Create the file system
- Create WebSphere Commerce Studio users
- Modify the publishing paths
- Modify the storesDocRoot
- Select the publishing method - ftp
- Publish the base store

13.2.2.1 Create the file system

Create a file system mount point on your AIX WebSphere Commerce Server. The file system created will be used as the target to publish your commerce application from studio. For detailed instructions on creating logical volumes and file systems on AIX refer to 11.1.2.4, "Create a logical volume and file system" on page 233.

1. Create a logical volume, for example, logical volume name: **projlv**
2. Create a file system on a previously defined logical volume, for example file system mount point: **/projects**.
3. Mount the file system, for example:

```
# mount /dev/projlv /projects
```

4. Allocate storage to the file system by typing the following:

```
# chfs -a size='200000' /projects
```

5. Change ownership of the file system by typing the following:

```
# chown -R db2inst1:db2iadm1 /projects  
# ls -l /projects
```

When publishing files from studio to the target host, the proper access permissions are needed for FTP, file system, and the database updates. This can be accomplished many different ways. Depending on the size of your organization, you may require different users. For the working example, we will make db2inst1, the database instance owner, the owner of the /projects file system by typing the following:

13.2.2.2 Modify the storesDocRoot

1. From the right-hand frame under the **Test** view, select and expand **<your_hostname>**, select **stores**, right-click -> **Properties**.
2. When the Properties window appears, select the **Publish** tab, then do the following:
 - Select checkbox **Publish this folder to a publishing target**
 - From the pull-down, select **storesDocRoot**
 - Deselect **Make this Folder a virtual directory**
 - Click **OK**

storesDocRoot

Anything that is added under the stores folder within WebSphere Commerce Studio will be published to storesDocRoot (default macro and images directories).

13.2.2.3 Select the publishing method - FTP

To select the publishing method, complete the following steps:

1. From the right-hand frame, select **<your_hostname>**, right click -> **Properties**.
2. When **<your_hostname>** Properties window appears, select the **Publishing** tab, then do the following:
 - Select **FTP publish** radial button
 - FTP Login: **<userid_instance_owner>** (for example db2inst1)

- Password: <password_instance_owner>
- Click **Save Password** checkbox
- FTP Port: 21 (default ftp port)
- Click **OK**

Note

You may see an error message stating that one or more of your publishing target paths are not safe. Ignore this message. We have additional configuration items to complete prior to publishing.

13.2.2.4 Modify publishing paths

Modify the publishing paths on the WebSphere Commerce Studio system by completing the following steps:

1. From the right-hand frame, select <your_full_qualified_hostname>, right click, select **Properties**.
2. When the Properties window appears, click **Define Publishing Targets**
3. When the Publishing Targets window appears, enter paths as seen in Figure 52. When you are finished, click **OK**.

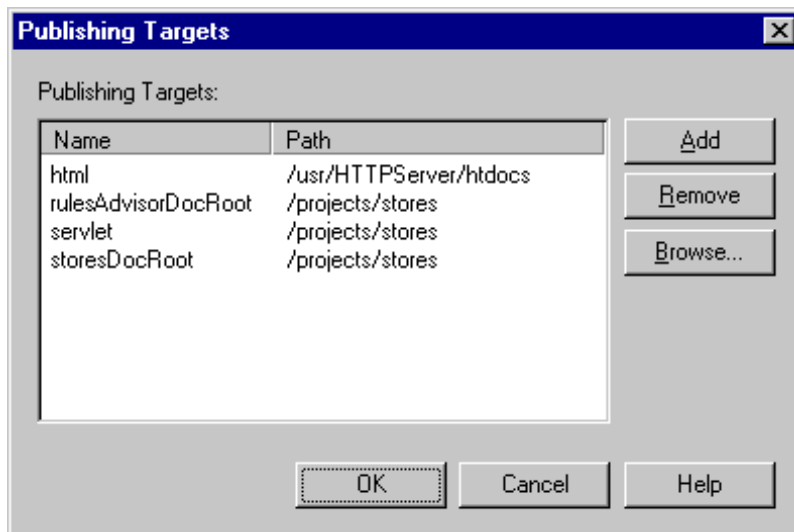


Figure 52. Target publishing path - AIX

Note

The HTML path is not being used in our example. If it was used, we would need to update the alias within the httpd.conf file for HTML doc root.

13.2.2.5 Publish the base store

To publish the base store to the target host, complete the following steps:

1. Verify that your WebSphere Application Server is started.
2. From the left-hand frame select **<your_project>** (itso) -> right click -> **Publish whole Project**.
3. When the Publishing Options window appears, click **OK**.
4. When the Files to publish window appears, click **OK**.
5. When the Commerce Suite server logon window appears, enter the following:
 - User name: ncdadmin
 - Password: <your_password>
 - Click **OK**

Note

The first time you publish you will get a message stating that folders need to be created. This will also happen if you create new folders to publish.

6. Change access permissions

On the target host commerce server, change the access permissions of the files published by typing the following:

```
# chmod -R 755 /projects
```

Note

You need to change the access permissions every time you publish your project to the AIX WebSphere Commerce server.

13.2.3 Configuring the WebSphere Commerce server

This section provides detailed instructions for configuring the WebSphere Commerce Suite for the published store.

13.2.3.1 Modify httpd.conf to add aliases

On your WebSphere Commerce server, you must update the httpd.conf file to include aliases and updated paths to your store. To modify the httpd.conf file, complete the following steps:

1. Stop the HTTP Server by typing the following:

```
# cd /usr/HTTPServer/bin
# ./apachectl stop
```

2. Change to the directory of the httpd.conf file:

```
# cd /usr/HTTPServer/conf
```

Windows NT

On a Windows NT system, the path is c:\ibm\http\conf.

3. Modify httpd.conf with a text editor:

- Search for the */stores* alias (comment out by using the #)

Note

The path on AIX is case sensitive. Verify the path of the published files on the commerce server.

- Insert the following aliases at the end of the commented WebSphere Commerce section:

```
Alias /itso "/projects/stores/ITSO"
Alias /ITSO "/projects/stores/ITSO"
Alias /stores "/projects/stores"
```

Note

We recommend that you add your aliases after the commerce aliases. When an instance is deleted, any alias entry within the commerce section is removed.

- Save and close the file

4. Start HTTP Server by typing the following:

```
# cd /usr/HTTPServer/bin
# ./apachectl start
```

13.2.3.2 Modify db2www.ini include and macro path

On your WebSphere Commerce server, you must update the db2www.ini file MACRO_PATH and INCLUDE_PATH to include the root directory of your publishing target. Insert your path at the beginning of the path statements. In the working example, c:\projects\stores is the root.

1. Change to the directory containing db2www.ini type the following:

```
# cd /usr/HTTPServer/htdocs/en_US
```

Windows NT

On a Windows NT system, the path is c:\ibm\http\htdocs.

2. Modify the db2www.ini file with a text editor. At the beginning of each line, insert the stores path:

```
MACRO_PATH <publishing_root_directory> (For example, /projects/stores)
INCLUDE_PATH <publishing_root_directory> (For example, /projects/stores)
```

3. Stop and start the WebSphere Commerce instance.

13.2.4 Verifying the base store

1. Start a browser, and enter the URL: `http://<your_hostname>/itso`
2. Verify the store:
 - Register a user
 - Log on
 - Browse/shop

13.3 Customizing a store

In this section we will customize the example store. For detailed instructions for many of the tasks, we refer to the *IBM WebSphere Commerce Suite Pro Edition, Commerce Suite Administrator V4.1* product guide.

13.3.0.1 Remote administration

To remotely administer the WebSphere Commerce Server for AIX, complete the following steps:

1. To launch the WebSphere Commerce Administrator, start the Netscape Communicator browser, and enter the following URL:
`http://<your_hostname>/ncadmin`
2. Log on

- User ID: ncadmin
- Password: <your_password>

13.3.0.2 Create shopper groups

Shopper groups are added by the Store Manager. In the working example, we need to create the following shopper groups:

- IBM
- Public
- Retail

For detailed instructions, refer to the Create Shopper Group section in the *IBM WebSphere Commerce Suite Pro Edition, Commerce Suite Administrator V4.1* product guide.

13.3.0.3 Enroll shoppers in shopper groups

Registered users are added to shopper groups by the Store Manager.

For detailed instructions, refer to the Enroll Shoppers in Shopper Groups section in the *IBM WebSphere Commerce Suite Pro Edition, Commerce Suite Administrator V4.1* product guide.

13.3.0.4 Create product categories

When creating product categories it is important to understand the hierarchy. We need to define a top level category under the ITSO store. In this example, we created a category called ITSO Categories under the ITSO store. Next we moved the sample category clothing within the ITSO Categories top level category. Finally, we created the secondary categories, first visible on the Web site, under the ITSO Categories category, as defined in Table 28.

Note

If your store has a hierarchy of categories, remember that you must have a top level category in the hierarchy.

Also, the top category needs a category template assignment to view other categories. This is described in 13.3.0.5, “Assign templates to categories” on page 285.

Table 28. ITSO category, subcategory, products, prices

Category	Sub Category	Product	Price
ITSO Categories			
Redbooks	WebSphere Redbooks	<i>WebSphere V3 Performance Tuning Guide, SG24-5657</i>	\$100
		<i>Servlet and JSP Programming with IBM WebSphere Studio and VisualAge for Java, SG24-5755</i>	\$100
	Patterns Redbooks	<i>e-Commerce Patterns using WebSphere Commerce Suite, Patterns for e-business, SG24-6156</i>	\$100
		<i>Patterns for e-business: User-to-Business Patterns for Topology 1 and 2 using WebSphere Advanced Edition, SG24-5864</i>	\$100
RedCDs	WebSphere CDs	WebSphere Vol 1 CD	\$10
		WebSphere Vol 2 CD	\$10
	Patterns CDs	Patterns Vol 1 CD	\$10
		Patterns Vol 2 CD	\$10

For detailed instructions, refer to the Create Product Categories Using the Commerce Suite Administrator section in the *IBM WebSphere Commerce Suite Pro Edition, Commerce Suite Administrator V4.1* product guide.

When complete your product category hierarchy should look like Figure 53.

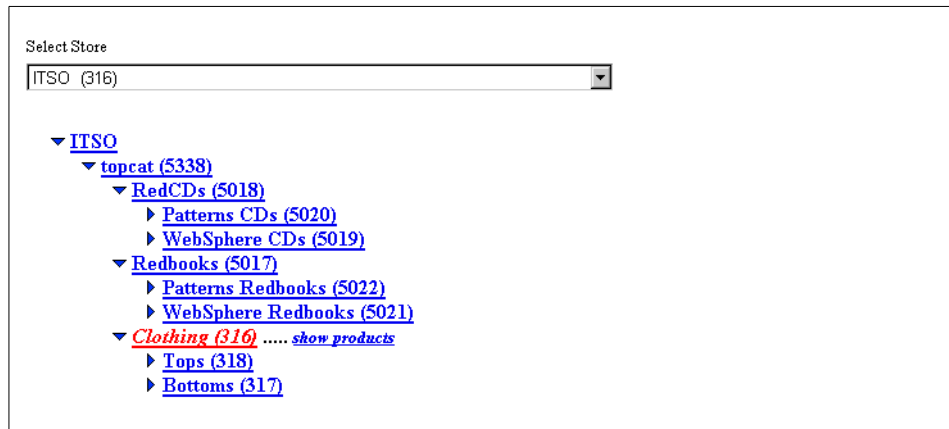


Figure 53. Category hierarchy for the ITSO store

13.3.0.5 Assign templates to categories

Categories in the hierarchy displaying other categories, and products need a template to display them. In this example, we used the template `/stores/macros/catdisp.d2w` to display other categories. As a reference, view the sample category templates (remember to click search to view the macro used by the template).

For detailed instructions, refer to the Assign templates to categories section in the *IBM WebSphere Commerce Suite Pro Edition, Commerce Suite Administrator V4.1* product guide.

13.3.0.6 Enter basic product information

For the working example, we need to add the products to the corresponding category as seen in Table 28.

For detailed instructions, refer to the Enter Basic Product Information section in the *IBM WebSphere Commerce Suite Pro Edition, Commerce Suite Administrator V4.1* product guide.

13.3.0.7 Enter basic product information

To display products, we need to assign a template. In this example, we used the `/stores/macros/proddisp.d2w` macro.

For detailed instructions, refer to the Assign templates to products section in the *IBM WebSphere Commerce Suite Pro Edition, Commerce Suite Administrator V4.1* product guide.

13.3.0.8 Enter prices

In the working example, we need to add product prices as seen in Table 28.

For detailed instructions, refer to the Enter Prices section in the *IBM WebSphere Commerce Suite Pro Edition, Commerce Suite Administrator V4.1* product guide.

13.3.0.9 Create discounts codes

In the working example, we need to create the discounts based on shoppers groups as follows:

- IBM - 75% discount
- Retail - 25% discount
- Public - no discount

For detailed instructions, refer to the Create Discount Codes section in the *IBM WebSphere Commerce Suite Pro Edition, Commerce Suite Administrator V4.1* product guide.

13.3.0.10 Change tax rates for a store

For detailed instructions, refer to the Change Tax Rates for a Store section in the *IBM WebSphere Commerce Suite Pro Edition, Commerce Suite Administrator V4.1* product guide.

13.3.0.11 Shipping services

For detailed instructions, refer to the Shipping Services section in the *IBM WebSphere Commerce Suite Pro Edition, Commerce Suite Administrator V4.1* product guide.

13.3.1 Add Category and Product data - massimport XML

The Mass Import utility can import data in either XML or ASCII delimited format. If you have large product catalogs this is the preferred method of data entry to the commerce database. Refer to the

By default, the Mass Import utility imports data to a category and product-related tables. However, it is possible to import data to any WebSphere Commerce Suite table. There are three mass import methods available in WebSphere Commerce Suite for DB2 database: SQL, DB2import and DB2load. For details on different import methods, please refer to the online help of WebSphere Commerce.

13.3.2 Verify the example store

After you have customized your store, you should verify that it is working properly before you proceed to the next chapter.

1. Publish the store

Refer to 13.2.2.5, “Publish the base store” on page 280.

2. Verify the store

Refer to 13.2.4, “Verifying the base store” on page 282.

Chapter 14. SecureWay Directory (LDAP) - AIX platform

As organizations realize the overhead and cost involved in managing many distributed micro and macro directories, the need for a centralized. IBM understands this requirement and supports it by providing an industry-standard directory implementation using the Lightweight Directory Access Protocol (LDAP) in the IBM SecureWay Directory product. The IBM SecureWay Directory product provides a general purpose directory that multiple applications can exploit, which is very important for a successful I/T operation in medium and large environments.

This chapter is organized into the following sections:

- Overview
- SecureWay Directory installation
- SecureWay Directory configuration
- SecureWay Directory Server Administration
- SecureWay Directory integration with WebSphere Commerce Suite
- SecureWay Directory Management Tool (DMT)
- SecureWay Directory SSL configuration
- Where to find more information

14.1 Overview

In this section we will provide an overview for the following topics:

- What is a directory?
- What is LDAP?
- Why should I use LDAP?
- IBM SecureWay Directory and WebSphere Commerce Suite

14.1.1 What is a directory?

A directory is a listing of information about objects arranged in some order that give details about each object. Common examples of this are a city telephone directory or a library card catalog. For a telephone directory, the objects listed are people and the details given about each person are addresses and telephone numbers. Directories allow users or applications to find resources that have the characteristics needed for a particular task. For example, a directory of users can be used to look up a person's e-mail

address or fax number. A directory of application servers could be searched to find a server that can access customer billing information.

14.1.2 What is LDAP?

The Lightweight Directory Access Protocol (LDAP) is an open industry standard that has evolved to meet these needs. LDAP defines a standard method for accessing and updating information in a directory. LDAP is gaining wide acceptance as the directory access method of the Internet and is therefore also becoming strategic within corporate intranets. It is being supported by a growing number of software vendors and is being incorporated into a growing number of applications. For example, the two most popular Web browsers, Netscape Navigator/Communicator and Microsoft Internet Explorer, support LDAP functionality as a base feature.

The LDAP information model is based on an entry, that contains information about some object, for example a person. Entries are composed of attributes, which have a type and one or more values. Each attribute has syntax that determines what kind of values are allowed in the attribute and how those values behave during directory operations.

14.1.3 Why should I use LDAP?

People and businesses are increasingly relying on networked computer systems to support distributed applications. These distributed applications might interact with computers on the same local area network (LAN), within a corporate intranet, within extranets linking up partners and suppliers, or anywhere on the worldwide Internet. Much of this information can be shared among many applications, but it must also be protected in order to prevent unauthorized modification or the disclosure of private information.

Information describing the various users, applications, files, printers, and other resources accessible from a network is often collected into a special database that is sometimes called a directory. As the number of different networks and applications has grown, the number of specialized directories of information has also grown, resulting in islands of information that are difficult to share and manage. If all of this information could be maintained and accessed in a consistent and controlled manner, it would provide a focal point for integrating a distributed environment into a consistent and seamless system. An LDAP directory provides a centralized repository of objects that are available from any place within the enterprise.

14.1.4 IBM SecureWay Directory and WebSphere Commerce Suite

The IBM SecureWay Directory product provides directory services for the WebSphere Commerce Suite. Many organizations require this level of security. WebSphere Commerce Suite can be configured to use LDAP for primary or secondary registration. In 14.5, “SecureWay Directory integration with WebSphere Commerce Suite” on page 308, we will provide instructions for configuration and an explanation of what services are offered.

Refer to 14.8, “Where to find more information” on page 326 for resource information about LDAP.

14.2 SecureWay Directory installation

In this section we provide detailed instructions on how to install IBM SecureWay Directory V3.1.1.2 for AIX, and prerequisite software.

14.2.1 Pre-installation requirements

For details on pre-installation requirements, refer to the SecureWay Directory online documentation system requirements found on the product CD.

For the example we used the following hardware and software, which meets the pre-installation requirements.

14.2.1.1 Hardware

We installed SecureWay Directory V3.1.1.2 for AIX on the following hardware:

- RS/6000 43P-140
- 512 MB RAM
- 4 GB hard disk

14.2.1.2 Software

We installed the following prerequisite software for the SecureWay Directory server:

- DB2 UDB Server V6.1.0.6 for AIX
- IBM JDK V1.1.8 for AIX
- IBM HTTP Server V1.3.6.2 for AIX

DB2 UDB Server V6.1.0.6 for AIX

The IBM SecureWay Directory data is stored in a DB2 database. During the SecureWay Directory installation process, the install program will create a

new DB2 instance and database, if you choose to create a default database during the installation. If you choose to use an existing database, you will need to provide information regarding the database, such as instance name, database name, DB2 administrator user, and password. Whether you choose to create the default database or use an existing one, remember that LDAPDB2 is a reserved name. If you use an existing database, the information contained in that database will be overwritten and lost (new schema, ldap data).

For detailed instructions for installing DB2, refer to *IBM WebSphere Commerce Suite Pro Edition for AIX, Installation Guide V4.1*, chapter 2 on page 11. Alternatively, you can refer to 11.1.3, “Install IBM DB2 UDB EE V6.1.0.6 for AIX” on page 235.

Note

When installing DB2 UDB V6.1.0.6 for AIX, you must install the DB2 Software Developers Kit as a prerequisite to SecureWay Directory. Failure to install this package will result in an installation prerequisite failure (db2_06_01.adt.rte 6.1.0.0 package missing) for the SecureWay Directory V3.1.1.2 product.

We were not able to find information about this issue in the SecureWay Directory Installation Guide, readme.txt, or addendum.txt.

IBM JDK V1.1.8 for AIX

For detailed instructions, refer to 11.2.3, “Install IBM JDK V1.1.8 for AIX” on page 241.

IBM HTTP Server V1.3.6.2 for AIX

For detailed instructions, refer to 11.2.4, “Install IBM HTTP Server (Apache) for AIX V1.3.6.2” on page 244.

If you do not intend to use SSL with SecureWay Directory, it is not necessary to configure the HTTP Server with SSL support. We recommend that you configure your SecureWay Directory Server with SSL support.

14.2.1.3 SecureWay Directory Server Administration

To perform administration tasks on the SecureWay Directory Server, a frame-enabled browser is required that supports the following:

- HTML Version 3.0 or later
- Java V1.1.8 features including JDK 1.1 AWT events

- JavaScript 1.2
- The browser must be enabled to accept cookies.

The following Web browsers support these specifications:

- Netscape Navigator 4.07 or higher (4.08 is recommended)
- Netscape Communicator 4.5 or higher

14.2.1.4 SSL support - GSKit

Global Security Kit (GSKit) is an optional software package that is required when using Secure Socket Layer (SSL).

Note

Although SSL support is optional, we strongly recommend that you install your server with SSL support. Without SSL enabled user ID's, and passwords will be passed from the WebSphere Commerce Suite Server to the SecureWay Directory Server.

SecureWay Directory V3.1.1.2 alone does not provide the capability for SSL connections from SecureWay Directory clients. The SSL feature is added by installing the IBM GSKit 3.01 package (located on the AIX Bonus Pack). The GSKit package includes SSL support and associated RSA (4) technology. There are two GSKit packages available on the respective AIX Bonus Packs: US and Export.

US Bonus Pack

The US Bonus Pack contains the following filesets:

- ldap.max_crypto_client
- ldap.max_crypto_server (pre-reqs ldap.max_crypto_client)
- gskru301.*

The gskru301 GSKit package is required to install the SSL support for use by the SecureWay Directory client and server (up to 128-bit and 3DES encryption). You should also install the ldap.max_crypto_server or ldap.max_crypto_client to upgrade the SecureWay Directory utilities and the Java Naming and Directory Interface (JNDI) support to use the higher levels of encryption. If you do not install the ldap.max_crypto_server or ldap.max_crypto_client, and only install the gskru301 package, then the SecureWay Directory filesets installed from the base Operating System CD provide a maximum of 56-bit DES encryption for SSL.

The US version is typically available only in the United States and Canada, but may be exportable under some circumstances to other countries.

Export Bonus Pack

The Export Bonus Pack contains the following filesets:

- ldap.exp_crypto_client
- gskre301.*

The gskre301 GSKit package is required to install SSL support for use by the SecureWay Directory client and server (up to 56-bit DES encryption). Install the ldap.exp_crypto_client only if you have Java applications that use the JNDI interface with SSL. By installing the gskre301 package, the SecureWay Directory filesets installed from the base Operating System CD are enabled to support a maximum of 56-bit DES encryption for SSL.

To install a secure SecureWay Directory from the SecureWay Directory package, first install the client or server from the package. Then the GSKit 3.01 can be installed from the AIX Bonus Pack.

Note

The SecureWay Directory server works without the GSKit installed. In this case it accepts only non-SSL connections from SecureWay Directory clients. Similarly, the SecureWay Directory client works without the GSKit installed.

14.2.2 IBM SecureWay Directory V3.1.1.2 for AIX installation

This section will guide you through the required steps to install the SecureWay Directory Server.

Note

It is not necessary to install security functions if you are not intending to use them. You can provide encryption by installing the Global Security Kit (GSKit), which is available on the CD Bonus Pack.

In this example, will install IBM SecureWay Directory V3.1.1.2 for AIX using smitty by completing the following steps:

1. Log in to AIX as user root and start an AIX terminal session.
2. Insert the SecureWay Directory CD into the CD-ROM drive.

3. Mount the CD-ROM:

```
# mount -r -v cdrfs /dev/cd0 /mnt
```

4. # cd /mnt/usr/sys/inst.images

5. # smitty install_all

6. When the Install and Update from All Available Software window appears, enter the following in the Input device/directory field:

```
./
```

Press Enter.

7. In the Software to Install field, press F4 for a list and highlight the following packages using F7 to select. Refer to Table 29 for filesets to install for the SecureWay Directory Server.

Table 29. SecureWay Directory Server filesets

Fileset	Description
ldap.client	SecureWay Directory client
ldap.html.<locale>	HTML install, config guides, and man pages. We selected ldap.html.en_US for U.S. English.
ldap.msg.<locale>	SecureWay Directory messages. We selected ldap.html.en_US for U.S. English.
ldap.server	SecureWay Directory server and administration interface.

8. When you finish selecting the filesets, press Enter.

9. When the Are You Sure? message appears, press Enter.

10. Check the installation summary at the end of the output to verify successful installation of the filesets.

11. Press F10 to return to a system prompt.

14.2.3 Install verification

To verify that SecureWay Directory was installed successfully type the following and compare the displayed results:

```
# ls|pp -l | grep ldap
```

```

ldap.client.adt          3.1.1.2 COMMITTED SecureWay Directory Client SDK
ldap.client.rte         3.1.1.2 COMMITTED SecureWay Directory Client
ldap.html.en_US.config  3.1.1.0 COMMITTED SecureWay Directory
ldap.html.en_US.man     3.1.1.0 COMMITTED SecureWay Directory Man Pages

```

ldap.msg.en_US	3.1.1.0	COMMITTED	SecureWay Directory Messages -
ldap.server.admin	3.1.1.2	COMMITTED	SecureWay Directory Server
ldap.server.com	3.1.1.2	COMMITTED	SecureWay Directory Server
ldap.server.rte	3.1.1.2	COMMITTED	SecureWay Directory Server
ldap.client.rte	3.1.1.2	COMMITTED	SecureWay Directory Client
ldap.server.admin	3.1.1.2	COMMITTED	SecureWay Directory Server
ldap.server.com	3.1.1.2	COMMITTED	SecureWay Directory Server

14.3 SecureWay Directory configuration

The SecureWay Directory server can be configured by using a graphical user interface (GUI) or by a command line utility.

14.3.1 Configure SecureWay Directory using a GUI

This section provides instructions for configuring SecureWay Directory.

14.3.1.1 Start the SecureWay Directory Configuration utility

This utility can be used to configure the following:

- Directory administrator distinguished name (DN), password
- Configure a database
- Configure a Web server

You can select one of these tasks or select multiple tasks. If you select more than one task, the information entry windows are displayed consecutively.

Note

If you already have a SecureWay Directory database make sure you select use existing database when prompted.

In the working example, we selected all three options. To configure a new SecureWay Directory system, complete the following steps:

1. Log in to AIX as user root and start an AIX terminal session.
2. Start the configuration utility by typing the following:

```
# ldapxcfg
```
3. When the SecureWay Directory Configuration welcome window appears, select the checkbox for each of the options as seen in Figure 54 on page 297 and then click **Next**.

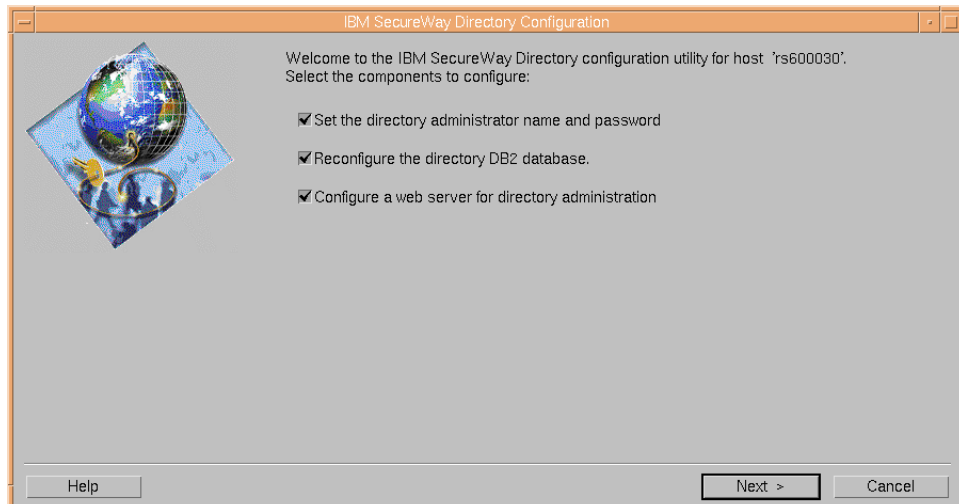


Figure 54. SecureWay Directory configuration - welcome

4. Enter the directory administrator distinguished name (DN), and password for your host window appears, enter the required information as seen in Figure 55, then click **Next**.

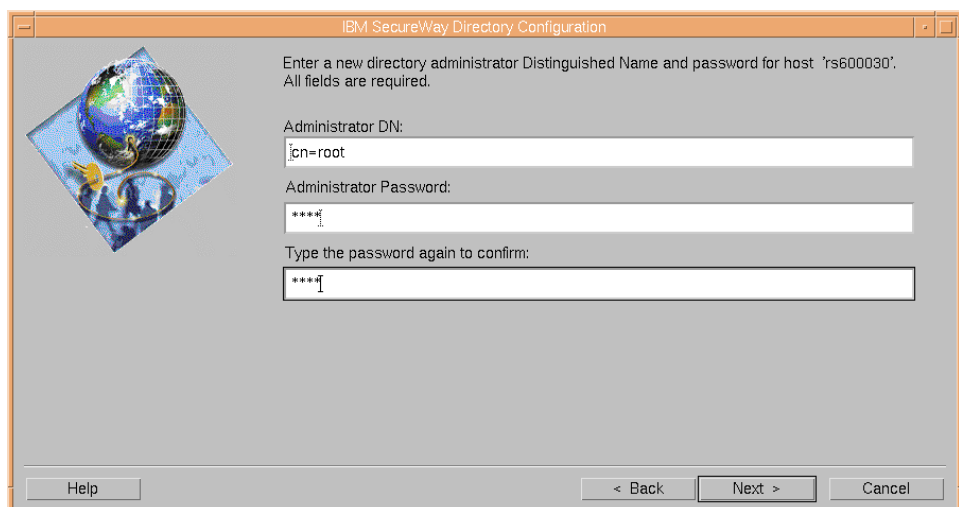


Figure 55. SecureWay Directory configuration - administrator DN and password

5. Configure DB2 database window appears as seen in Figure 56. Based on the following options make your selection, then click **Next**.

- New LDAP database - select **Create a default LDAPDB2 database**. For the working example we chose this option.
- Existing database - select **Use my own DB2 database**

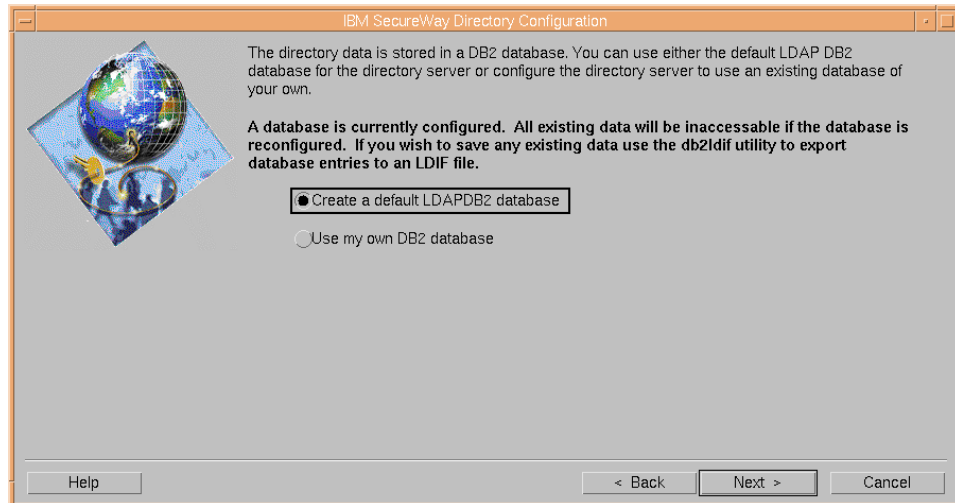


Figure 56. SecureWay Directory configuration - database configuration

6. Create the database using locale codepage or universal character set (UCS-2) window appears, select **Create the default DB2 database** as seen in Figure 57 on page 299, then click **Next**.

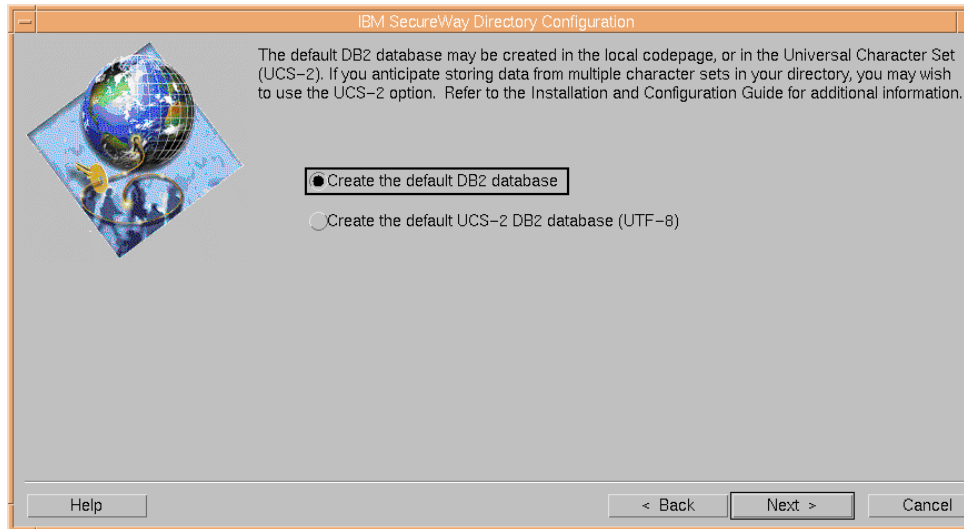


Figure 57. SecureWay Directory configuration - create DB using code page or UCS

7. Enter the database directory path to store the LDAPDB2 database as seen in Figure 58, then click **Next**.

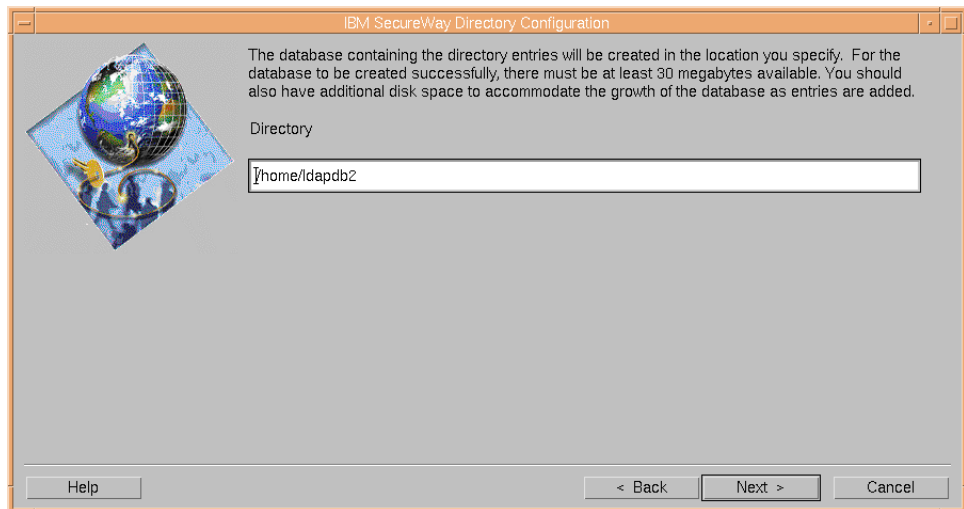


Figure 58. SecureWay Directory configuration - database directory path

8. Select a Web server for directory administration window appears, select **IBM HTTP** as seen in Figure 59, then click **Next**.

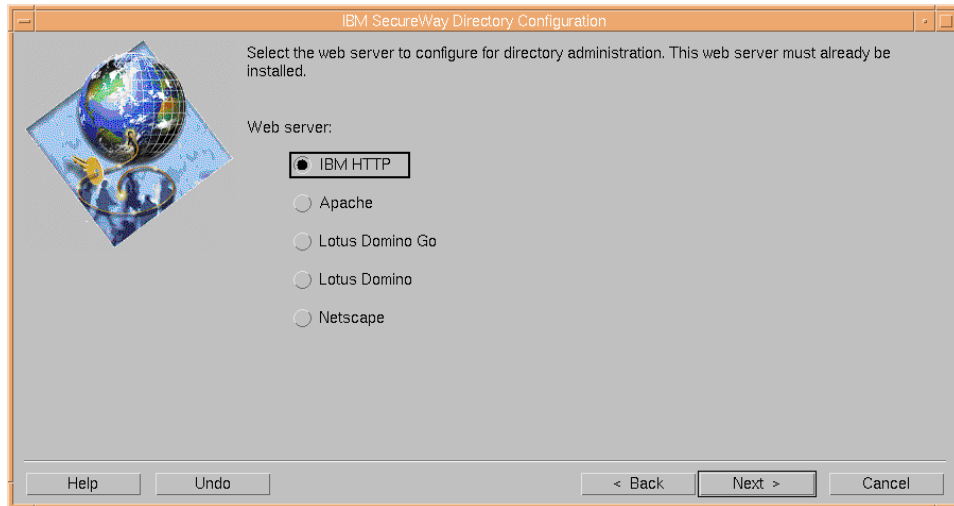


Figure 59. SecureWay Directory configuration - Web server directory administration

9. Enter the full path name of the Web server configuration file window appears as seen in Figure 60. For the IBM HTTP Server V1.3.6.2 enter the following:

`/usr/HTTPServer/conf/httpd.conf`

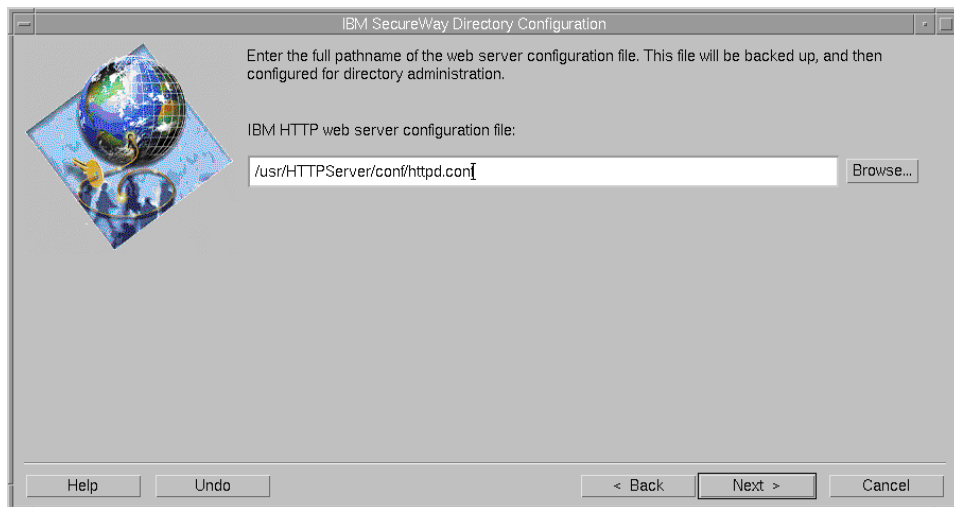


Figure 60. SecureWay Directory configuration - HTTP Server configuration file path

10. When the Review the Configuration Summary window appears, click **Configure**.

11. When the Configuration Completion window appears, make a note of the administration URL, then click **OK**.
12. Restart the configured Web server. For the IBM HTTP Server V1.3.6.2 type the following:

```
# cd /usr/HTTPServer/bin
# ./apachectl restart
```
13. To start the SecureWay Directory Server from the command line, type the following:

```
# cd /usr/ldap/bin
# ./slapd
```

Note

The spelling of the command above is correct (slapd).

14.4 SecureWay Directory Server Administration

The SecureWay Directory Server Administration GUI is a Web-based administration utility that supports the following administrative tasks:

- Start and stop the server
- Configure a database
- Configure a replica
- Add and delete suffixes
- Add entries to the directory database
- Define an ACL

14.4.1 Start and stop the server

To start the SecureWay Directory Server Administration GUI, complete the following steps:

1. Start a Web browser, enter URL: `http://<hostname>/ldap`
2. SecureWay Directory Server Administration logon window appears, as seen in Figure 61 on page 302, enter the following:
 - User ID: `cn=root`
 - Password: `<your_password>`
 - Click **Logon**



Figure 61. Logon for the SecureWay Directory Server Administration GUI

3. From the left hand frame, select **Server -> Startup/Shutdown -> Startup**
4. When the startup is finished, a completion window is displayed.

14.4.2 Add and delete suffixes

Before entries can be added to an LDAP directory, a *suffix* must be defined for that directory. A suffix specifies the distinguished name (DN) for the root of a directory, and is the highest entry stored in the directory for a server. Each entry stored ends with this suffix, and each entry to be added to the directory must have a suffix that matches one of the defined suffixes on the server. An LDAP server can house multiple suffixes, which can be added and deleted from the directory.

14.4.2.1 Add a suffix

To add a suffix to a directory, complete the following steps:

1. Start the SecureWay Directory Administration GUI
 - Start a Web browser, enter URL: `http://<hostname>/1dap`
2. SecureWay Directory Administration logon window appears, as seen in Figure 61, enter the following:
 - User ID: `cn=root`
 - Password: `<your_password>`

- Click **Logon**
- 3. From the left navigation frame, click on **Suffixes**.
- 4. Click **Add a suffix**.
- 5. Add a suffix for this server frame appears, enter the following:
 - Suffix DN: o=ibm,c=us (working example)
- 6. Click n **Add a new suffix**.
- 7. Restart the SecureWay Directory Server:
 - From the left hand frame of the Directory Server Administration GUI, click the upper right icon next to the question sign.

14.4.2.2 Delete a suffix

To delete a suffix from a directory, complete the following steps:

1. Click **Delete suffixes**
2. Click the box next to the suffix you want to delete.
3. Click the **Delete suffixes** button.
4. To restart the SecureWay Directory Server:
 - From the left hand frame of the Directory Server Administration GUI, click the upper right icon next to the question sign.
5. You can verify your operation by clicking **Suffixes** -> **List suffixes** as seen in Figure 62.

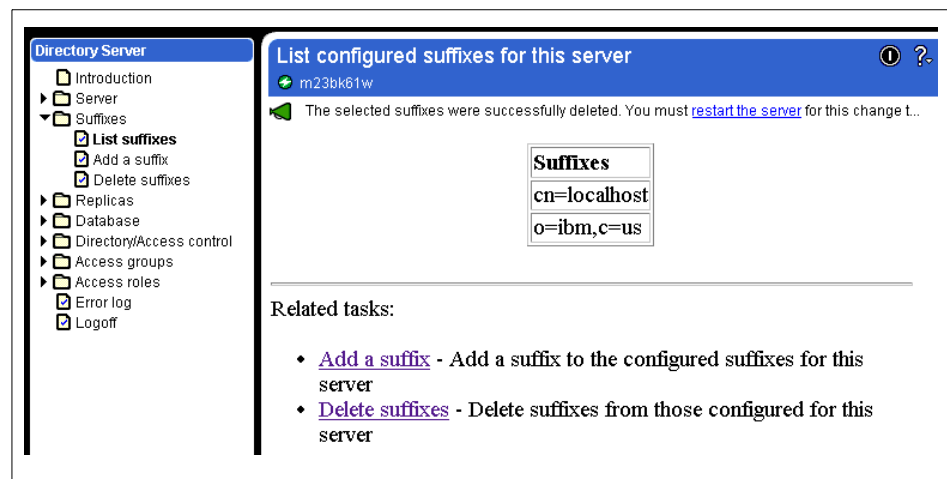


Figure 62. Listing Suffixes - restarting the server

Note

Removing a suffix will disable access to all directory data underneath that suffix. This does not physically remove the data from the directory. Access can be restored to a deleted suffix by adding it again. The suffixes are recorded in the slapd.conf file (we do not recommend that you manually edit this file).

14.4.3 Add entries to the directory database

There are two methods of adding entries to the directory database:

1. LDIF (LDAP Data Interchange Format) data file

LDIF is a standard format for representing LDAP entries in text form and is used to import entries into the database.

2. SecureWay Directory Management Tool (DMT)

The SecureWay Directory Management Tool provides a GUI to add entries to the database. For more information, refer to 14.7, “SecureWay Directory Management Tool (DMT)” on page 317.

Note

Prior to adding entries to the database a suffix (DN) must exist on the server, and all entries to be added must have a suffix that matches the DN value.

14.4.3.1 Adding data entries to the database using LDIF

There are two ways to import an LDIF file, either through the SecureWay Directory Server Administration GUI or by using command line utility. For the working example we have provided step-by-step instructions for using the SecureWay Directory Server Administration GUI to import the LDIF file.

LDIF import - SecureWay Directory Server Administration GUI

For the example below, the sample LDIF file shipped with the product was used. Some editing is usually necessary to adapt the LDIF file to your environment (for example, to define the correct suffix).

To add data entries to the database using LDIF with a GUI, complete the following steps:

1. Start the SecureWay Directory Server Administration GUI, and log on.
2. From the left navigation frame, click **Database -> Add entries**.

3. Add entries to the database from an LDIF file frame appears, specify an LDIF file. In this example, we used the ITS0.LDIF file.

Note

The LDAP user ID must have proper AIX permissions on the LDIF file prior to the import into the database in the next step.

This file has been created manually for demonstration purposes. The content of the ITS0.LDIF file are as follows:

```
version: 1
```

```
dn: CN=LOCALHOST
cn: localhost
objectclass: container
objectclass: top
aclsource: default
ownersource: default
aclpropagate: TRUE
ownerpropagate: TRUE
entryowner: access-id:CN=ROOT
aclentry: group:CN=ANYBODY:normal:rsc
```

```
dn: o=IBM, c=US
objectclass: top
objectclass: organization
o: IBM
```

```
dn: ou=ITS0, o=IBM, c=US
ou: ITS0
objectclass: top
objectclass: organizationalUnit
seealso: cn=Hernan Cunico, ou=ITS0, o=IBM, c=US
```

```
dn: cn=hcunico,ou=ITS0,o=ibm,c=us
givenname: Herman
sn: Cunico
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
uid: hcunico
mail: hcunico@ar.ibm.com
cn: hcunico
```

4. Enter the location of the file containing the entries in the text entry field and click **Add entries to database**.

This loads the LDIF file into the directory. If successful, you should see the message below:

The directory was successfully updated.

Note

Remember that o=IBM,c=US suffix must be added before importing this directory data.

5. Restart the SecureWay Directory Server
 - From the left hand frame of the Directory Server Administration GUI, click the upper right icon next to the question sign.

14.4.3.2 Verify the data entries

There are several ways to verify the data entries.

- Scan the error log

Scanning the error log is a good way to verify whether or not an update has been successful. Remember that an LDIF file may contain a large number of entries, and errors may occur on just single entries of such a bulk load or update. For this reason, you should always check the error log.
- Directory Management Tool

To view data entries using the DMT, refer to 14.7, “SecureWay Directory Management Tool (DMT)” on page 317.
- SecureWay Directory Server Administration GUI
 - a. Start Web browser, enter URL: `http://<fully_qualified_host_name>/ldap`
 - b. Log on.
 - c. Click **Directories/Access control -> Browse tree**.
 - d. Select and expand the suffix you want to display.
 - e. Select and expand the top directory you want to display.
 - f. Click any suffix and a new window will appear showing the attributes for that suffix.

If the update was successful the directory entries will look similar to what is displayed in Figure 63, and Figure 64.

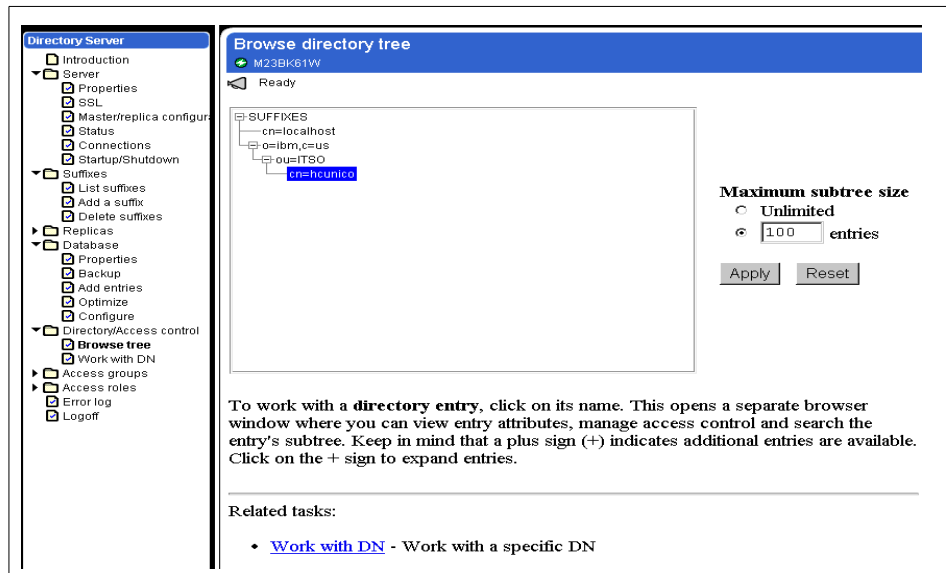


Figure 63. Browse directory tree - using Web based GUI

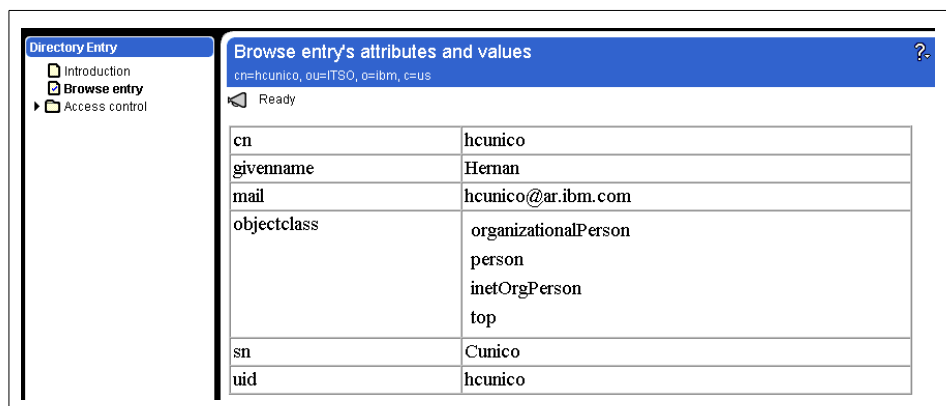


Figure 64. Browse entry's attributes and values - using Web based GUI

Note

We have configured SecureWay Directory to work with **inetOrgPerson** objectClass as seen in Figure 64.

14.5 SecureWay Directory integration with WebSphere Commerce Suite

This section provides instructions for configuring WebSphere Commerce Suite to use SecureWay Directory (LDAP). Table 30 provides information on the LDAP services offered to help determine whether to use the primary or secondary registry for the WebSphere Commerce Suite.

Table 30. Comparison of LDAP support level

	LDAP as Primary Registry	LDAP Secondary Registry	No LDAP Support
Shopper Information			
Shopper Authentication	Shopper information stored in the LDAP server is trusted.	Shopper information stored on the Commerce Suite database is trusted.	Shopper information stored on the Commerce Suite database is trusted.
Basic Authentication (shopper logs on to a Commerce Suite store)	The shopper is authenticated against the LDAP server using the Logon ID (LDAP UID) and password provided by the shopper. Shopper information on the LDAP server is replicated to the Commerce Suite database for run-time operational use.	The shopper is authenticated against the Commerce Suite database using the Logon ID and password provided by the shopper.	The shopper is authenticated against the Commerce Suite database using the Logon ID and password provided by the shopper.
Shopper Registration			
New shopper (registers on a Commerce Suite registration page)	Shopper information is stored both on both the Commerce Suite and LDAP servers.	Shopper information is stored in both the Commerce Suite database and the LDAP server.	Shopper information is stored in the Commerce Suite database.

	LDAP as Primary Registry	LDAP Secondary Registry	No LDAP Support
Existing shopper (updates registration information)	Shopper information is stored on both the Commerce Suite database and the LDAP server.	Shopper information is stored in the Commerce Suite database and on LDAP server.	Shopper information is stored in the Commerce Suite database.
LDAP user registers or updates information from another LDAP application	The information is replicated only when the user logs on to Commerce Suite. The user is authenticated against the LDAP server. Upon authentication information replicated to Commerce Suite database.	User information is not replicated to Commerce Suite. Since shopper information is replicated from Commerce Suite to LDAP. This user will not be recognized as a Commerce Suite shopper since authentication is performed against local Commerce Suite database.	N/A
Availability of LDAP server	LDAP server must always be available. Commerce Suite caches shopper information for run-time operational use.	LDAP server does not always have to be available. Shopper information replication from Commerce Suite to LDAP is not guaranteed, but performed at best effort.	N/A
LDAP connection fails during shopper registration or logon	An error page is displayed to the shopper. An error message is logged in the system log file.	No error page is displayed to shopper. An error message is logged to the log file.	N/A

14.5.1 Configure WebSphere Commerce instance

To configure your WebSphere Commerce instance with support for LDAP, complete the following steps:

1. Ensure that the SecureWay Directory Server is started.
2. On the WebSphere Commerce Server, start the Configuration Manager.

```
# cd /usr/lpp/CommerceSuite/server/bin
# ./start_config
```

3. When the Configuration Manager logon window appears, enter the following:
 - User ID: webadmin (default)
 - Password: <your_password> (default webibm)
4. Select your Websphere Commerce instance and then click **Settings**, then click the **LDAP** tab.
5. Enter LDAP instance configuration settings as seen Figure 65 on page 310, then click **OK**. Enter the hostname of the SecureWay Directory Server. By default the host field entry will contain the WCS hostname.

Commerce Suite Server	Caching	Instance Data	Web Server	Database
Payment	LDAP	Rule Server	Rule Service	

LDAP Type: Primary

Domino Single Logon:

Host: M23BK61Vf.itso.ra1.ibm.com

Port: 389

Administrator DN: cn=root

Administrator Password: ****

Confirm Password: ****

LDAP Search Root: ou=itso,o=ibm,c=us

Shopper Base DN: ou=itso,o=ibm,c=us

Shopper Objectclass: top person organizationalPerson i

Character Set: No translation to UTF-8

OK Cancel Help

Figure 65. WebSphere Commerce Suite Configuration - LDAP Settings

Table 31 provides a detailed explanation about each field, how it is used and what for. This will help you to have a better understanding of LDAP.

Table 31. Configuration Manager Fields - LDAP

LDAP Type	Select whether you want to use LDAP as a secondary registry, as primary registry, or not at all. For more information on which level of LDAP support you require, see Table 30. Primary Registry in our example
Domino Single Logon	This checkbox is only valid when LDAP is selected as a primary registry. Enable this checkbox to allow Domino users to log on to Commerce Suite with their existing ID.
Host	The fully qualified hostname specifying where the LDAP server is installed.
Port	The port used by the LDAP server. 389 in our example
Administrator DN	The distinguished name of the Administrator. <i>cn=root</i> in our example
Administrator Password	The LDAP administrator's password.
Confirm Password	Re-enter the LDAP administrator's password.
LDAP Search Root	The distinguished name of all the search roots, separated by semicolons (;). ou=itso,o=ibm,c=us in our example
Shopper Base DN	The distinguished name of default base to store new shoppers registered from Commerce Suite. ou=itso,o=ibm,c=us in our example
Shopper Objectclass	The objectclass used to store shopper objects on the LDAP server, separated by a space. top person organizationalPerson inetOrgPerson in our example
Local character set for data translation into UTF-8 by LDAP	Local character set for data translation into UTF-8 format by LDAP. The LDAP server must support UTF-8 database if a character set is chosen. No translation to UTF-8 in our example

6. WebSphere Commerce will update WebSphere Commerce Server instance, Web server configuration, and prepare the database. When finished click **OK**.
7. Verify that the instance is **Active**, and if not click **Start**.

8. Start the SecureWay Directory Server.

Note

When the WebSphere Commerce Suite instance is modified, (deleted, created) and subsequently stop/started, we had to restart the SecureWay Directory Server manually. Incidentally, you may need to update your aliases in the httpd.conf file, and update the db2www.ini file.

We have now completed the installation and configuration steps. Next we will verify the configuration.

14.5.2 Verify the configuration

After you have completed all of the installation and configuration steps we need to verify the SecureWay Directory Server and WebSphere Commerce Suite implementation by completing the following steps:

1. Start a Web browser, enter URL:

http://<fully_qualified_host_name>/<your_store>

Note

In this example, we used the store created in Chapter 13, “Create and publish a store” on page 273. Alternatively, you can use the demomall.

2. When the Home page appears, click on **Register**.
3. When the New Registration page appears, fill-in the highlighted fields.

Note

When you are registering as a new user, and have configured LDAP as a primary registry in your WebSphere Commerce Server instance, be sure that you fill in all the highlighted fields and *First Name*. If not, you will receive an error message like shown in Figure 66.



Sorry, problem found using user registry,
Please contact the Site Administrator for help.

Problem Description:

**User registry temporarily unavailable, please try
again later.**

Figure 66. Error message when a required field during registration is missing

4. Check Table 32 to verify the required fields for each product.

Table 32. Required fields for WebSphere Commerce and SecureWay Directory

Required Fields	WebSphere Commerce	SecureWay Directory
Shopper's Login ID	X	X
Password	X	X
Verify Password	X	
Last Name	X	X
First Name		X
Address (line 1)	X	
City	X	
Zip/Postal Code	X	
Country	X	

5. You may repeat previous step many times as needed. However, between registrations you must close all Web browser windows and start all over again. When you register, a cookie remains in memory to maintain a

session between the WebSphere Commerce Server and the Web browser client.

6. Start the Directory Management Tool (DMT)
 - a. Move to the SecureWay Directory Server machine to start the DMT.
 - b. Log in to AIX as user root and start an AIX terminal session.
 - c. Start the DMT by typing the following:

```
# dmt
```
7. When the Directory Management Tool window appears, the default view is Browse the directory tree, if not, choose **Tree -> Browse tree**.
8. Depending on the number of users registered, you should see something similar to Figure 67.

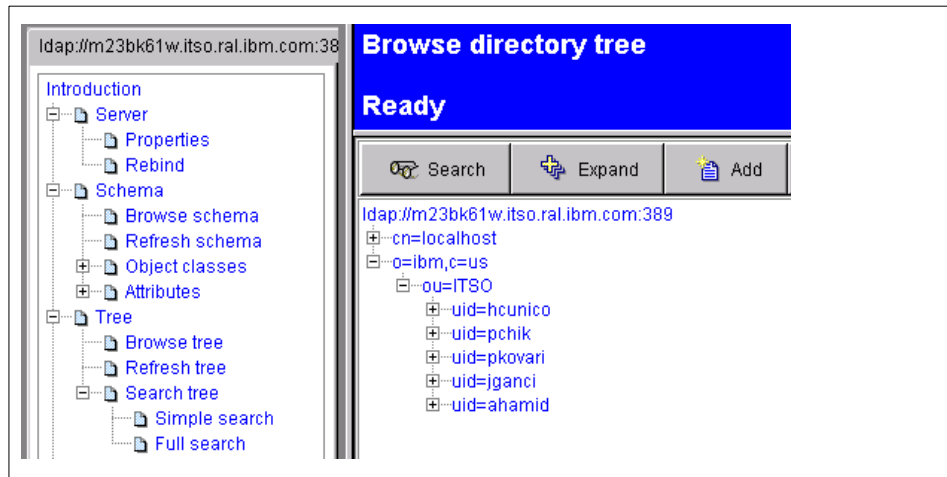


Figure 67. Browse directory tree - Directory Management Tool

14.6 SecureWay Directory SSL configuration

When using SecureWay Directory with WebSphere Commerce Suite, we recommend that you configure SSL (Secure Sockets Layer) in order to encrypt the communication between SecureWay Directory Server and WebSphere Commerce Server.

Note

Although SSL support is optional, we strongly recommend that you install your server with SSL support. Without SSL enabled user ID's, and passwords will be passed from the WebSphere Commerce Suite Server to the SecureWay Directory Server.

14.6.1 Prerequisites

- GSKit - SSL configuration

Refer to 14.2.1.4, "SSL support - GSKit" on page 293

- IBM HTTP Server - SSL configuration

The IBM HTTP Server must be configured for SSL prior to the SecureWay Directory SSL configuration. Refer to 11.2.4.5, "Enabling SSL for the HTTP Server" on page 248 for detailed instructions.

14.6.2 SecureWay Directory SSL configuration

To configure SecureWay Directory SSL, complete the following steps:

1. Start a Web browser, enter URL: `http://<full_qualified_host_name>/ldap`
2. Log on to IBM SecureWay window appear, enter the following:
 - User ID: `cn=root` (distinguished name)
 - Password: `<your_password>`
 - Click **Logon**
3. From the left frame, select **Server -> SSL**
4. When the Server SSL (Secure Sockets Layer) properties window appears, enter the required fields as seen in Figure 68, then click **Apply**.

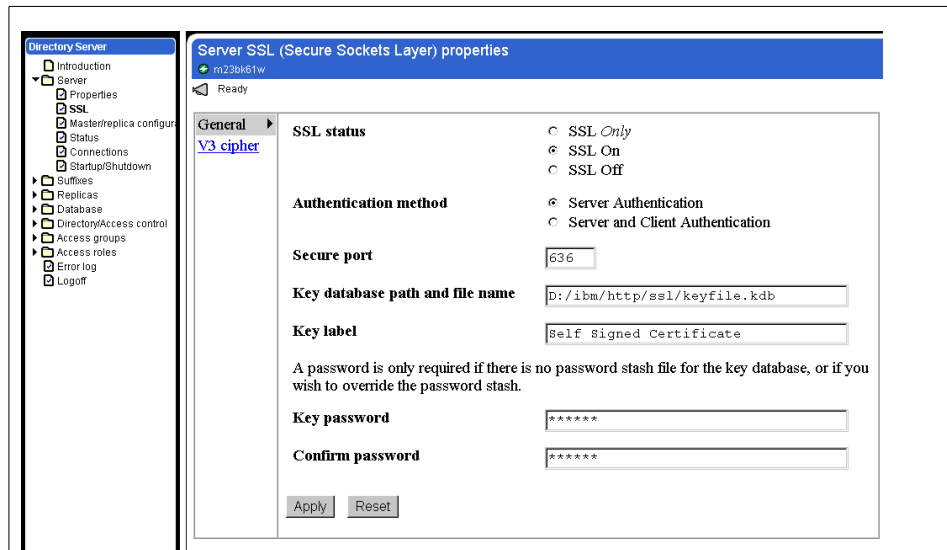


Figure 68. Server SSL (Secure Sockets Layer) properties

Note

The key database path, file name, key label, key password must match the configured values of the IBM HTTP Server. Refer to 11.2.4.5, “Enabling SSL for the HTTP Server” on page 248.

5. Restart the SecureWay Directory Server
 - From the left hand frame of the Directory Server Administration GUI, click the upper right icon next to the question sign.
6. The SecureWay Directory Server can now communicate in both SSL and non-SSL modes with the WebSphere Commerce Server.

14.6.3 Security Verification

Complete the following steps to verify the SSL configuration:

1. Start Netscape Communicator browser.
2. In the Communicator pull-down menu, click **Address book**.
3. The **Address book** window appear, click **File -> New Directory**.
4. When the Directory Server Property window appears, fill-in the required fields as seen in Figure 69, then click **OK**.

At this point you can choose either **secure** or **standard** connection and verify both. The port number will change automatically.

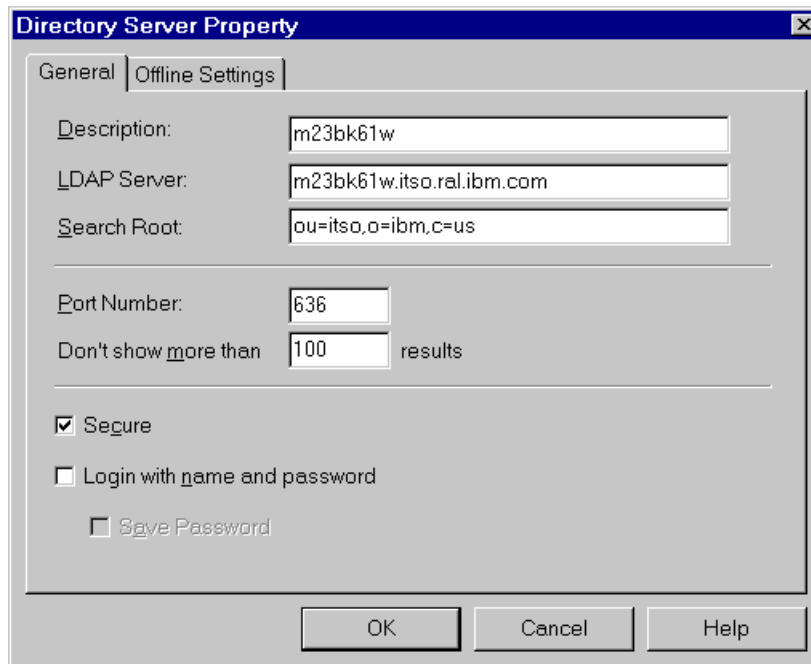


Figure 69. Directory Server Property window

5. Select the new directory you just created, and click the **Search** button in the upper right corner.
6. When the Basic Search window appears, enter * in the name field, and click **Search**.
7. A list showing all users should be visible in the right hand frame. In our example, we have entered only one user.

This concludes the Security verification. We have successfully installed and configured the SecureWay Directory Server for use with WebSphere Commerce Suite.

14.7 SecureWay Directory Management Tool (DMT)

The IBM SecureWay Directory Management Tool (DMT) provides a graphical user interface that enables you to manage information stored in directory servers. The tool can be used for the following tasks:

- Connecting to directory servers
- Displaying server properties
- Administering schema object classes and attributes
- Administering directory entries
- Administering directory entry ACLs

LDAP is a client/server protocol for accessing a directory service. It was initially used as a front-end to X.500, but can also be used with stand-alone and other kinds of directory servers. LDAP can be used as a centralized information repository to support information sharing among various clients.

14.7.1 Connecting to directory servers

This section provides detailed instructions for connecting to SecureWay Directory servers.

14.7.1.1 Server log on

If a directory user DN and password are not provided in the DMT configuration file, the tool connects as an anonymous user once it is started. Although an anonymous user can browse the directory tree and schema, to perform directory updates, in most instances, you need to log on as a directory user. To modify the directory server schema you must log on as the server administrator.

To log on as a different user complete the following steps:

1. Start the Directory Management Tool (DMT):
 - a. Move to the SecureWay Directory Server machine to start the DMT.
 - b. Log in to AIX as user root and start an AIX terminal session.
 - c. Start the DMT by typing the following:


```
# dmt
```
2. Click **Server -> Rebind**.
3. Provide the user DN and password, click Enter.

The DMT uses the `/usr/lldap/etc/dmt.conf` configuration file. We have provided a sample `dmt.conf` file below:

```
#browser=
server1.url=ldap://rs600030.itso.ra1.ibm.com:389
#server1.security.bindDN=
#server1.security.password=
#server1.security.ssl.keyclass=
```



```
#server1.security.ssl.keyclass.password=
```

In this file you can specify LDAP servers URL, port, user DN, password, keyclass file name, and password for SSL connections.

Note

We strongly recommend for security reasons, that you do not provide DN user name and password in the dmt.conf file.

14.7.1.2 Connect to an additional server

If localhost is provided in the URL of the configuration file, DMT will connect to the LDAP directory server on the local machine where DMT is running. To add a connection to another directory server, complete the following steps:

1. From the DMT, click **Add server**.
2. Provide the server name, port, user DN, and password.
3. Press Enter.

Note

Do not specify the URL prefix (for example: ldap://) in the server name.

You can choose to connect to the server via SSL by first selecting **Use SSL** and then providing the keyclass file name and password.

14.7.2 Display server properties

You can view the server properties by clicking **Server->Properties**. The current bind DN, subschema entry, supported LDAP protocol versions, and the naming contexts that the server holds are displayed.

14.7.3 Administering schema object classes and attributes

This section provides information for browsing, adding, editing, and deleting schema attributes and object classes.

14.7.3.1 Browse directory schema

IBM SecureWay Directory V3.1.1 provides dynamically extensible schema support. An administrator can define new attributes and object classes to enhance the default schema. The directory schema can be browsed and updated with the DMT. You must log on as a directory administrator to update the schema.

To browse the directory schema complete the following steps:

1. From the DMT, click **Schema** -> **Browse schema**.
2. Click the + sign to display the following selections:
 - Attribute types
View all defined attributes, or alternatively click **Schema** -> **Attributes** -> **View attributes**.
 - Object classes
View all defined object classes, or alternatively click on **Schema** -> **Object classes** -> **View object classes**
 - Syntaxes
To view all supported syntaxes.
 - Matching Rules
View all supported matching rules.

14.7.3.2 Add an object class

To add an object class, complete the following steps:

1. From the DMT, click **Schema** -> **Object classes** -> **Add object class**.
2. Provide the object class name, description, and a unique string of object identifier (the OID).
3. Select the superior object class from which to inherit attributes.
4. Determine the object class type. (The default is structural).
5. Click the **Required attributes** tab:
 - Select the **MUST have attributes** from the attribute list in the left window.
 - Click **Add** to move the selection to the right window.
 - You can also select the attributes in the right window and then click **Remove** to deselect them.
 - Click **OK**.
6. Click the **Optional attributes** tab, and then select the **MAY have attributes**.
7. Click **OK**.

14.7.3.3 Edit an object class

This operation is similar to add an object class except that a pull-down menu is provided for the selection of the object class to be edited. To edit an existing object class, complete the following steps:

1. Click on **Schema -> Object classes -> Edit object classes**.
2. Select an object class to be edited from the pull-down menu.
3. Provide the description.
4. Click the **Required attributes** tab and select the MUST have attributes.
5. Click **OK**.
6. Click the **Optional attributes** tab and then select the MAY have attributes.
7. Click **OK**.

14.7.3.4 Delete object classes

To delete one or more object classes complete the following steps:

1. Click **Schema -> Object classes -> Delete object classes**.
2. Select the object class or classes to be deleted.
3. Click **Delete**.
4. Click **OK** to confirm the deletion.

14.7.3.5 Add an attribute

To add an attribute complete the following steps:

1. Click **Schema -> Attributes -> Add attributes**.
2. Provide an attribute name, description, and an OID.
3. Select a syntax for this attribute from the list.
4. Determine whether this is a multi-valued attribute.
5. Select the matching rules used.
6. Click **OK**.

Note

Advanced users can click the IBM extensions tab to change the DB2 table name, the DB2 column name, the security class, and the indexing. We recommend indexing an attribute to optimize performance for LDAP searches.

14.7.3.6 Edit an attribute

This operation is similar to add an attribute process except that a pull-down menu is provided for the selection of the attribute to be edited. To edit an existing attribute complete the following steps:

1. Click **Schema -> Attributes -> Edit attributes**.
2. Select an attribute from the list.
3. Make the necessary entries in the General screen, then Click **OK**.

Note

Advanced users can click the IBM extensions tab to change the DB2 table name, the DB2 column name, the security class, and the indexing. We recommend indexing an attribute to optimize performance for LDAP searches.

14.7.3.7 Delete attributes

To delete one or more object classes complete the following steps:

1. Click **Schema -> Attributes -> Delete attributes**.
2. Select the attribute or attributes to be deleted.
3. Click **Delete**.
4. Click **OK** to confirm the deletion.

14.7.4 Administering a directory tree

This section provides information for browsing, searching, adding, editing, and deleting schema attributes and object classes.

14.7.4.1 Browsing a directory tree

You can browse the directory tree by using the Browse tree option. When you browse the directory tree, the directory contents are displayed according to the directory hierarchies. To open part of the tree, expand the entries. The entries that are in the next level down are displayed. To browse the directory tree complete the following steps:

1. Click **Tree -> Browse tree**.
2. To expand the tree one level, click on a + sign.

The tool bar at the top of the window allows for an operation on a selected entry in the tree to be initiated. The operations include, Add, Search, Edit, Delete, Expand, ACL settings, and Edit RDN. Use **Edit** to view an entry.

When an entry (a node in the tree) is selected, click the **Expand** button to expand the entire subtree below the entry.

Double-click an entry to edit it.

14.7.4.2 Searching a directory tree

To allow convenient access to entries of special object classes such as user and group, DMT provides a simple search option in addition to a full search option. While the full search option allows you to provide complete specifications of all parameters to a directory search operation, the simple search requires only minimal input for searching through a set of entries that belong to a selected object class.

Simple search

To perform a simple search, complete the following steps:

1. Click **Tree** -> **Search tree** -> **Simple search**.
2. Select the type of entry to search.
3. Determine the filter for the search result.
4. Click **OK**.

Full search

To perform a full search, complete the following steps:

1. Click **Tree** -> **Search tree** -> **Full search**.
2. Input the search constraint:
 - Search base DN (the default is all suffixes)
 - Scope (the default is subtree)
 - Size limit (the default is unlimited)
 - Time limit (the default is unlimited)
 - Alias de-referencing (the default is no)
 - Referral chasing (the default is yes)
3. Click the **Search filter** tab on the top of the display.
4. Input the search filter. If necessary, use the AND or OR connectors.
5. Click the **Search return set** tab.
6. Select the attributes to be returned or the full entry.
7. Click **OK**.

14.7.4.3 Adding an entry

To add an entry to the directory tree, complete the following steps:

1. Click **Entries** -> **Add entry**. You can also add an entry if you click **Browse tree**, select the parent entry, and then click **Add** on the toolbar.
2. Provide the parent DN and the Relative Distinguished Name (RDN) for the new entry. The RDN must be entered as an attribute=value pair.
3. Choose the object type (object class) from the list or **other** for more options. If **other** is selected you can specify either a structural object class or an auxiliary object class or both.
4. Click **OK**.
5. Another screen displays the attributes associated with the selected object class. Highlighted fields are required fields. Enter the attribute values for the entry. Use the edit icon to add multiple values.

When the action is initiated from the tree browsing screen, the parent entry can be selected from the directory tree and the parent DN is entered automatically.

Note

The flyover that appears when the cursor is positioned over the attribute name, or the text field describes the syntax of that attribute.

14.7.4.4 Edit an entry (or view an entry)

To view an entry, complete the following steps:

1. Click **Entries** -> **Edit entry**. You can also select an entry if you click on **Browse tree** and then double-click the entry.
2. Provide the entry DN to edit or view.
3. Click **OK**.
4. Another screen displays the attributes associated with the selected object class. Highlighted fields are required fields. Enter the attribute values for the entry. Use the edit icon to add multiple values.

Like adding an entry, this operation can be launched from the browsing tree window. The entry can be edited from the tree by double-clicking it.

14.7.4.5 Delete an entry

To delete an entry from the directory tree complete the following steps:

1. Click **Entries** -> **Delete entry**, or from the browsing window click the entry to be deleted.
2. Click **Delete**.
3. Click **OK** to confirm the deletion.

14.7.4.6 Edit an entry RDN

To edit an entry RDN, complete the following steps:

1. Click **Entries** -> **Edit entry RDN**. You can also select an entry if you click **Browse tree** and then click the entry.
2. Enter the current DN and provide the new RDN for the entry. (You can change the new RDN; the current DN is not editable).
3. Click **OK**.

14.7.5 Administering directory entry ACLs

To modify an ACL for an entry complete the following steps:

1. Click **Entries** -> **ACLs**. Enter the DN and then click **OK**. You can also select the entry if you click on **Browse tree** and then click on the **ACL** button on the top of the window.
2. On the **ACLs** tab:
 - Either edit the existing list or create a new subject ACL list for a new subject.
 - Determine whether descendant entries are to inherit the ACL lists.
 - Mark the **Remove** check boxes for those subject ACL list which are to be removed.
 - Mark/unmark the check boxes for the rights to add/remove.
 - Mark/unmark the check boxes for the rights to read/write/search/compare the attributes of three security classes.
 - To add a new subject ACL list, enter the subject DN, select the subject type, and then click **ADD**. A new list is added and ready for changes.
3. Click the **Owners** tab.
 - Determine whether descendant entries are to inherit the owner list.
 - Mark the **Remove** check boxes to remove owners.
 - To add a new owner, enter the owner DN, select the owner type, and then click **ADD**.
4. Click **Change**.

14.7.6 Troubleshooting

This section provides some trouble shooting tips the first time you edit a suffix or add an entry to a suffix:

An error occurred getting attributes for entry c=us: noSuchObject.

This means that the suffix contains no data.

To add data to a suffix

1. In the navigation menu click **Entries** -> **Add entry**.
2. Leave the Parent DN blank and specify the suffix as the entry (for example c=us).
3. Select the object type (for example Country) and click on **OK**.
4. Fill in the desired attribute values and click on **Create**.

Rebind a suffix

You should now be able to edit the suffix as well as add entries. You can find the bind DN in either of two ways from the menu area:

1. You can locate it on the **Rebind to Server** panel. Click on **Server** -> **Rebind** to display the panel. The rebind DN will be displayed in the **User DN** field. (If the bind DN is anonymous, the **Anonymous** radio button is checked.)
2. Click on **Server** -> **Properties**. In the table, under Server attributes, find the bind dn property.

14.8 Where to find more information

- *LDAP Implementation Cookbook*, SG24-5110
- *Understanding LDAP*, SG24-4986
- *Application Server Solution Guide, Enterprise Edition: Getting Started*, SG24-5320
- General IBM SecureWay Directory information:
<http://www.ibm.com/software/network/directory/>.

Chapter 15. Java technologies in WebSphere Commerce Suite

Version 4.1 of the WebSphere Commerce Suite and WebSphere Commerce Studio include support for JavaServer Pages, Java Servlets, and JavaBeans technology. The foundation for hosting these Java technologies is the WebSphere Application Server, Advanced Edition.

This chapter is organized into the following sections:

- Overview of Java technologies in WebSphere Commerce Suite
- Configuring the WebSphere Application Server
- JSP and bean example
- Custom servlet and bean example
- Where to find more information

15.1 Overview of Java technologies in WebSphere Commerce Suite

This section provides an overview of JavaServer Pages, Java Servlets, JavaBeans technology and how they are used in the WebSphere Commerce Suite and WebSphere Commerce Studio.

15.1.1 JavaServer Page (JSP)

JavaServer Page (JSP) technology allows you to create HTML pages that include dynamic elements. As a result, you can use a JSP to create catalog pages, which typically contain dynamic content, such as products, product prices, and attributes. The JSPs are used in WebSphere Commerce Suite to display categories, product lists, and products.

Commerce Studio includes a set of Commerce Suite beans that you can drag and drop anywhere on a catalog page created with JSP technology. These beans allow you to retrieve category, merchant, and product information from the database without writing any code. You can also add your own images, static text, tables and other elements using Page Designer's WYSIWYG page editing function without any prior programming knowledge.

When a customer requests a JSP, a JSP-enabled engine from the WebSphere Application Server interprets the JSP tags and scriptlets, creates the content in the form of an HTML page, and returns it to the browser. Using JSPs to create dynamic HTML pages allows you to separate the task of coding data retrieval from the task of creating the display format. For example

a Java programmer can develop Java beans to retrieve data from the database while a media specialist can design the look and feel of the page.

Templates are used within the Commerce Suite to implement category and product assignments. Category templates can be assigned to different shopper groups using the Category Template Assignment form. You can allow certain shoppers to view categories with information and page styles directed specifically to them. The WebSphere Commerce Suite provides the following methods for implementing catalog templates:

- JSPs in Commerce Studio Page Designer
- Net.Data macro in a text editor
- Modify the supplied sample product template

We will focus on the JavaServer Page technology method of creating catalog templates for categories, product lists, and products.

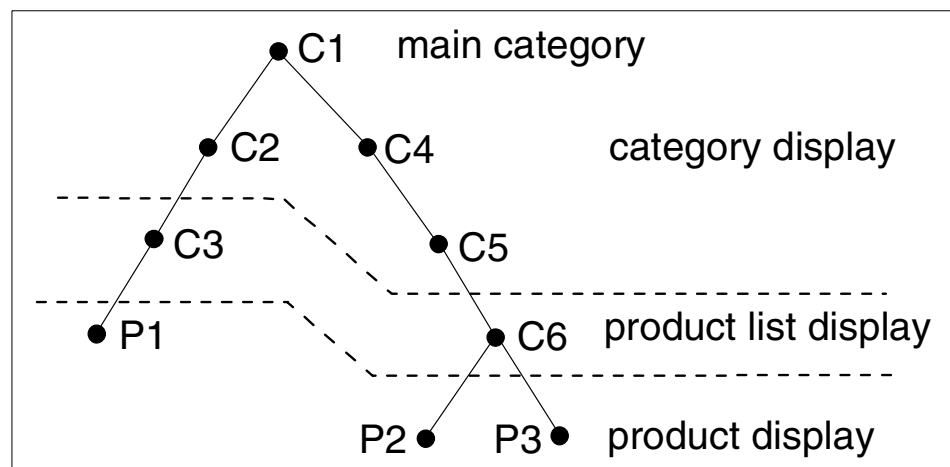


Figure 70. Category and product tree

Figure 70 presents a simple view of the store catalog. The *C* nodes represent the categories, while the *P* nodes represent the products. It is important to understand the following issues presented in Figure 70:

- Catalog structure
- Display pages
- How does a Net.Data display page work?
- How does a JSP display page work?

15.1.1.1 Catalog structure

The catalog display page templates are assigned to categories (C1, C2, C4, C5). In order to drill down we need to have references to the pages (C1->C2, C2->C3, C1->C4, C4->C5, C5->C6) where a category links to another category. At the lowest level of the categories (C3, C6), we need to reference the product list page. The lowest level branch displays products (P1, P2, P3).

15.1.1.2 Display pages

The display pages differ depending on the level in the catalog structure. Category display pages present other categories (C1, C2, C4, C5). Product list pages are the lowest categories (C3, C6). At the lowest level in the hierarchy are the product display pages used to present products (P1, P2, P3).

Note

JSP and Net.Data display pages cannot be mixed. References from a JSP cannot refer to a Net.Data display page. Net.Data display pages cannot refer to JSPs.

15.1.1.3 How does a Net.Data display page work?

When a store is created by the Store Creator, the category display pages and product display pages use catdisp.d2w. In catdisp.d2w, both type of pages are bundled in one file.

Category display

To display a category, the macro accesses the category table in the database, and lists all the subcategories that belong to the current category. In this case, the list of categories are displayed. Next the macro accesses the products table in the database and tries to list all the products that belong to the current category. In this case, there are child categories and no products, so the product list is blank.

Product list display

To display a product list, the macro accesses the category table in your database and lists all the subcategories that belong to the current category. In this case there are no subcategories, so the list is blank. Next the macro accesses the products table in your database, and tries to list all the products that belong to the current category. In this case the products list is displayed.

Note

Technically the macro works for both category and product display; however, this is not a good approach. The macro accesses the database two times from one page.

15.1.1.4 How does a JSP display page work?

When using JSPs for display pages you need to implement a category display JSP, product list JSP, and product list JSP. Each of these display types are detailed in the JSP example and require only one access to the database from each page.

One of the most powerful features of JSPs is that you can access beans and EJBs from within a JSP. Beans can be used from a class file, a serialized bean, a bean that is dynamically generated by a servlet, or a servlet itself.

Beans allow you to do the following:

- Create a bean from a serialized file or a class file.
- Refer to a bean from an HTTP session.
- Pass a bean from the servlet to the JSP.

You can access reusable server-side components simply by declaring the components in the JSP. Bean properties can then be accessed inside the file.

Dynamic content generation works in the following way:

1. The user fills in an HTML form, and clicks Submit. This will post the request to a Java servlet.
2. The servlet reads the input parameters and passes the parameters to JavaBeans that perform the business logic.
3. Based on the outcome of the business logic and the user profile, the servlet calls a JSP page to present the results.
4. The JSP page extracts the results from the JavaBeans and merges them with the HTML page. The dynamically generated HTML page is returned to the user.

The product list page displays the product page features of one particular product in an online store. It typically includes a description, price, and an image, and if the product has variations (for example, different sizes and colors) it allows shoppers to choose a variation. The page also contains a button for shoppers to add the selection to their interest list or shopping cart.

You can create product pages by developing a template that automatically generates the product pages.

There are several factors that determine the template design and the number of templates you need to create. You should first consider the types of products you are selling. For products that are available only in one form (no variation in size, color, and so forth), you should design a template that includes a paragraph describing the product in detail, one or more graphics, and a price.

For products that are available in more than one form, provide a description of the product's general qualities and features on the template, and then below it, include a section for the attributes (sizes, colors, and so forth). For example, to design a template for a v-neck shirt that comes in four sizes and three colors, you would first describe its general characteristics (such as price) and then include the seven attributes so shoppers can choose the appropriate combination of size and color. Each combination of attributes is referred to as an item, which shoppers can then add to their interest list or shopping cart.

Information about items typically appears below the product's general description, in the format of a form that shoppers can fill out. You can also create a separate template for the items.

A product template typically contains the following elements:

- Product name
- Full-sized picture
- Detailed description
- Price (For some products, it may not be possible to display the product price at this point, since some attributes may change the value of the price. For example, a 50 ft. garden hose costs less than a 100 ft. one.)
- A form to display item attributes, with a button to add the selected item to the interest list or shopping cart
- Header (optional)
- Navigational footer (optional)

The number of templates you choose to create will depend on how many different page layouts you want. For example, you may want to create a separate template for each group of related products in each store, or just one template for all products. You may also want to display products differently to different shopper groups. You can assign different product

templates to different shopper groups, using the Product Template Assignment form. This way, you can allow certain shoppers to view products with information and page styles directed specifically to them. For example, a shopper purchasing computer systems for a large company will most likely want different product information from a shopper purchasing a computer for home use.

15.1.2 Java Servlets

Similar to the way applets run on a browser and extend the browser's capabilities, HTTP servlets run on a Java-enabled Web server and extend the Web server's capabilities. Servlets are Java programs that use the Java Servlet Application Programming Interface (API) and the associated classes and methods. In addition to the Java Servlet API, servlets can use Java class packages that extend the API.

HTTP servlets extend the Web server capabilities by creating a framework for providing request and response services over the Web. When a client sends a request to the server, the server can send the request information to a servlet and have the servlet construct the client response.

A servlet can be loaded automatically when the application is loaded, or it can be loaded the first time a client requests its services. After loading, a servlet continues to run, waiting for additional client requests. By using servlet aliases (servlet URLs), multiple instances of a servlet can be loaded with a different instance for each alias.

Servlets perform a wide range of functions. For example, a servlet can:

- Create and return an entire HTML page containing dynamic content based on the nature of the client request.
- Create a portion of an HTML page (an HTML fragment) that can be embedded in an existing HTML page.
- Communicate with other server resources, including databases and Java-based applications.
- Communicate with other servlets. For example, you can use the WebSphere Administrative Console to define a servlet filter (a sequence of servlets, also known as a servlet chain).
- Filter data by MIME type for special processing, such as image conversion and server-side includes (SSI).
- Handle connections with multiple clients, accepting input from and broadcasting results to the multiple clients. A servlet can be a multi-player game server, for example.

A servlet is a Java program that plugs into a Web server. A Web server can be extended to host servlets through a servlet engine. Servlets make it easy to expand from client/single-server applications to multi-tier applications. Servlets allow businesses to connect databases to the Web.

Servlets greatly improve portability. Because servlets are written in Java, they are portable across platforms; they do not have to be recompiled for different operating systems. The servlet interface is a standard, so servlets can be moved from one servlet engine to another, as long as the servlets do not use vendor extensions. However, even if vendor extensions are used, the servlet engine will support a variety of Web servers, which means that the servlet will not be locked into a single platform. Consequently, programmers can develop on an operating system that has good tool support, such as Windows NT, and then deploy on an operating system with good scalability, such as AIX.

You can develop, debug, and deploy servlets within the VisualAge for Java Integrated Development Environment (IDE). In the IDE, you can set breakpoints within servlet objects, and step through code to make changes that are dynamically folded into the running servlet on a running server, without having to restart each time.

Although a servlet can be a completely self-contained program, the task of generating dynamic content should be split into the following two parts, to ease server-side programming:

- The business logic (content generation), which governs the relationship between input, processing, and output
- The presentation logic (content presentation, or graphic design rules), which determines how information is presented to the user

In this scenario, business logic can be handled by beans, and presentation logic can be handled by JSPs, while the servlet handles the HTTP protocol. With JSPs, you can efficiently separate the business logic of an application from its presentation logic.

When a JSP-generated servlet code is imported into the VisualAge for Java IDE, the following occurs:

1. The JSP source is fed to a page compiler, which creates an executable object (for example, a Java HTTP servlet).
2. VisualAge for Java then imports the generated servlet code. You can run and debug the servlet by using your browser to call the JSP that created the servlet.

15.1.3 JavaBeans

The WebSphere Commerce Studio includes a set of JavaBeans, called Commerce Suite beans, that access information from the tables in the database. Creating store pages using these beans allows you to display information that may change often, such as category and product lists, or product prices. Typically you will use Commerce Suite beans when creating pages for your catalog, such as category, product list, or product pages.

Each Commerce Suite bean accesses information from a particular table. For example, the Category bean accesses information from the Category table. In that table, the bean accesses information from a particular column through properties. That is, the category reference number property accesses the information in the CGRFNBR (category reference number) column in the category table.

Although each Commerce Suite bean contains distinct properties, they all share the following properties:

- Merchant reference number - the merchant reference number associated with the bean.
- User reference number - the user reference number associated with the request of the bean.
- Error message - the error message for the bean. The message is available if the request to activate this bean fails.
- Error code - the error code for the bean. The code is available if the request to activate this bean fails.

Some Commerce Suite beans work in conjunction with others. For example, the Child Category List bean will return a list of the applicable Category beans within a particular catalog, and the Product List bean will return a list of the applicable Product beans for a particular category within a particular catalog. The Product bean includes the Product Attribute List bean and the Price bean, allowing you to display the attributes and price for that product.

Using Page Designer, you can add the Commerce Suite bean of your choice to your store page, and select which properties you want to display from the Bean Property Selection window.

Note

Although the WebSphere Commerce Suite supports templates created with JavaServer Pages (JSP) technology or Net.Data, you can only use one type in your store. That is, if you create a product page template using JSP technology, you must create category page and product list templates with JSP technology.

The Commerce Suite includes the following beans:

1. Child category list bean

The child category list bean returns a list of category beans. Use the child category list bean to display a list of subcategories for the current category in the catalog.

2. Category bean

Use the category bean to display information about a particular category.

3. Product list bean

The product list bean returns a list of Product beans. Use the Product List bean to display a list of products within a category in the catalog.

4. Product bean

The product bean lists all product attributes. It returns a list of Product Attribute List beans, which each list a product attribute, for example, color or size. Use the Product bean to display information about a product.

5. Product attribute list bean

The product attribute list bean lists all values for a given attribute. It returns a list of Product Attribute beans which each associate a value to an attribute, for example red, green or blue for color, or S, M, or L for size.

6. Product attribute bean

The product attribute bean associates a single value with a single attribute, for example S for size, or blue for color.

7. Price bean

Use the price bean to display the price of a product in the catalog.

8. Merchant bean

Use the merchant bean to display information about the merchant (for example, the company address, and phone number).

9. Dynamic data source beans

Dynamic data sources beans are used to render complex elements on Product Advisor display pages, such as tables, lists or trees.

For more information on Commerce Suite beans refer to the online help at the following URL: http://<CommerceStudio dir>\html\en_US\nchehelp\index.htm then follow the links: **Concepts** -> **Commerce Suite Beans**.

WebSphere Commerce Suite beans are located in <WebSphere Commerce Suite dir>\CommerceStudio\databaseans\lib\wcsclient.jar file.

Also you have to know, that these beans, packaged into the JAR file, used by the Commerce Server, so do not use them in your own development, unless you have exact information about this.

15.2 Configuring the WebSphere Application Server

This section provides detailed instructions for configuring the WebSphere Application Server for the published store when using JSPs, servlets, and beans.

1. Launch the WebSphere Application Server Administrator's Console.
2. Select **Topology** tab.
3. Select and expand **WebSphere Administrative Domain**.
4. Select and expand **<your_hostname>**.
5. Select and expand **WebSphere Commerce Server**.
6. Select and expand **WCS Servlet Engine**.
7. Select **WCSServlets** Web application.
8. From the right-hand frame, select the **General** tab, and enter the following:

Web Application Web path: /webapp/commerce (default)

9. Select the **Advanced** tab, and enter the following:
 - Document Root: /projects/stores/ITSO
 - Add Classpath entries for the following:
 - /projects/stores/ITSO/servlets
 - /projects/stores/ITSO/jsp
 - /usr/lpp/CommerceSuite/classes (default)

Note

For Windows NT, enter the following classpath entries:

```
c:\projects\stores\itso\servlets  
c:\projects\stores\itso\jsp  
c:\ibm\wcs\classes (default)
```

10. Stop/Start the Application Server - WebSphere Commerce Server from the WebSphere Administrator's Console.

15.3 JSP and bean example

The JSP and bean example provides step-by-step instructions for implementing, testing, and deploying JSPs in WebSphere Commerce Suite. This section is organized into the following topics:

- Creating a JSP category list page template
- Creating a JSP product list page template
- Creating a JSP product page template
- Deploying JSPs
- Testing JSPs
- JSP development recommendations

15.3.1 Creating a JSP category list page template

This section provides step-by-step instructions for creating a JSP category list page within the WebSphere Commerce Studio. In order to complete the following example you need to have a store project within the WebSphere Commerce Studio. To create a store refer to Chapter 13, "Create and publish a store" on page 273.

Creating a JSP category list page template includes the following high-level steps:

1. Create a JSP file.
2. Add the `childCategoryListBean` to the JSP file.
3. Add dynamic category information to a JSP file.
4. Create links to the subcategory pages.

15.3.1.1 Create a JSP file

To create a JSP file, complete the following steps:

1. Start WebSphere Commerce Studio and open your project.
2. From the left-hand panel, select and expand <your_project> folder.
3. Select the **jsp** folder within <your_store> folder, and then right-click and select **Insert -> File**.
4. When the Insert File window appears, click **Create New** tab. Select **Blank.jsp**, and then enter the file name `catdisplay.jsp`.

Note

To rename a file in WebSphere Commerce Studio, select the file from the left-hand panel, and then select **Edit -> Rename** from the menu bar.

5. Click **OK**. The `catdisplay.jsp` file should be listed within <your_store> jsp folder as seen in Figure 71.

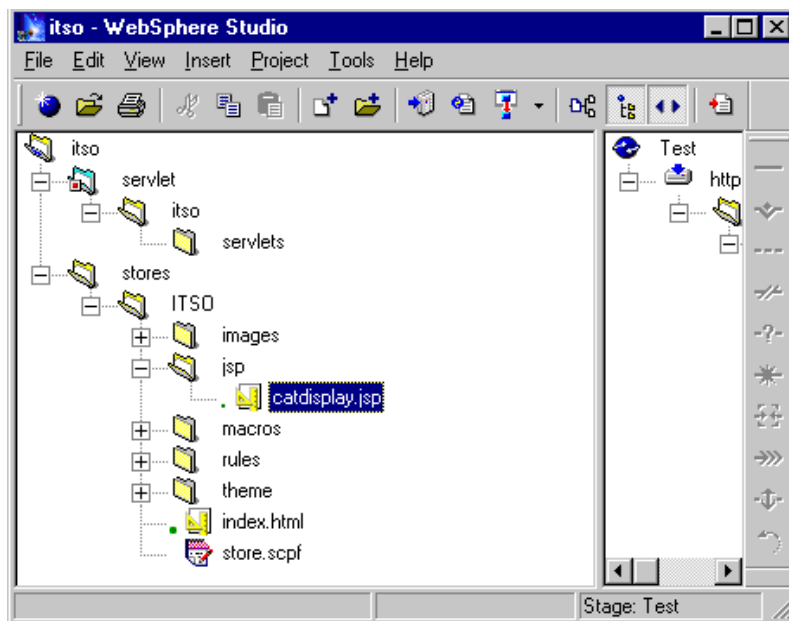


Figure 71. Commerce Studio - jsp folder file insert

15.3.1.2 Add the ChildCategoryListBean to the JSP file

Adding the ChildCategoryListBean to the JSP file will allow you access to the category information in the WebSphere Commerce Suite database.

To add a childCategoryListBean to a JSP, complete the following steps:

1. Double-click **catdisplay.dsp** to open the file in Page Designer.
2. Select the **Normal** tab (default), which is displayed at the bottom of the Page Designer panel.
3. From the Page Designer **View** menu, ensure that the **Contents Manager** is selected.
4. In the Contents panel, the left panel of the Page Designer, navigate to the <drive>:\<path_CommerceStudio>\databeans\lib folder.
5. From the drop down file list, select **Java Bean Files (.class; .jar)**.
6. Select **wcsclient.jar**, and then drag and drop the file into the Normal view.

Note

The wcsclient.jar contains the WebSphere Commerce beans for the business logic processes.

7. When the Bean Selection window appears, select **com.ibm.commerce.beans.ChildCategoryListBean**, and then click **OK**.
8. When the Attribute window appears, select the **jsp:useBean** tab (default).
9. Select **request** from the Scope drop-down list.

Note

The tag is used to set the attributes of the bean.

10. Select the **setProperty** tab, enter the following:
 - Property: * (type an * in this field)
 - Click **Add**

Note

Adding the * sets all properties in the bean that match the parameters in the page URL to the values specified in the URL. In this instance, the values for the merchant reference number (merchant_rn) and the category reference number (cgrfnb) are set in the bean.

11. Click **OK**. The {J} icon should be blinking in the Page Designer window.
12. If you are creating a category page template, your next step is to add dynamic category information.

15.3.1.3 Add dynamic category information to a JSP file

If you are creating a category page template, you will want to add category information to your page dynamically from the Commerce Suite database by using the `childCategoryListBean`.

To add dynamic category information, complete the following steps:

1. From the menu bar, select **Insert -> Dynamic Elements -> Dynamic Loop**.
2. When the Attribute window appears, click **Browse**.
3. When the Bean Property Selection window appears, select and expand the **childCategoryListBean**.
4. Select **List of child categories[]**, and then click **OK**.
5. The `childCategoryListBean.childCategory[]` should be displayed in the Loop Property field of the Attribute window. Click **OK**.

Note

This procedure generates inline Java code into the JSP, that parses through the vector called `child categories[]`. You can view the code under the HTML tab in Page Designer.

6. The loop property box should appear next to the {J} icon. Place the cursor on the Loop Property box.

Note

As you complete the following steps, ensure the cursor displays the hash mark, which denotes the loop property box.

7. Most category pages include the following:

- Category name
- Thumb nail image

Add category name

To add a category name, complete the steps:

1. From the menu bar, select **Insert -> Dynamic Elements -> Property Display**.
2. When the Attribute window appears, click **Browse**.
3. When the Bean Property Selection window appears, select and expand **childCategoryListBean**.

4. Select and expand **List of child categories[]**. Select **Category name**, and then click **OK**.
5. When the Attribute window appears, enter the following:
 - Sample Text: `CATEGORY_NAME`
 - Click **OK**.

The sample text is displayed in the category page. When the category page is requested, the sample text will be replaced with information from the Commerce Suite database.

Note

By setting the properties of the bean, the JSP can get the properties by using the get method at runtime. The JSP tag `<%= %>` can be found under HTML Source view, within Page Designer.

Add thumbnail image

To add a thumbnail image, complete the following steps:

1. From the menu bar, select **Insert -> Dynamic Elements -> Dynamic Image**.
2. When the Attribute window appears, select **Dynamic** tab (default), click **Browse** next to SRC Attribute.
3. When the Bean Property Selection window appears, select and expand the **childCategoryListBean**.
4. Select and expand **List of child categories[]**, select **Thumbnail image path name**, and then click **OK**.
5. When the Attribute window appears, the `childCategoryListBean.childCategory[].thumbnailPathName` should be displayed in the **SRC Attribute** field. Click **OK**.

The sample image should be displayed on the category page. When the category page is requested, the sample image will be replaced with the image from the Commerce Suite database for the category requested.

6. To further customize your pages, use the features in Page Designer, or add beans of your choice.
7. Select **File -> Save** to save your work.

15.3.1.4 Create links to subcategory pages

Once you have created the category pages, you must add links to the subcategory pages.

To create links to subcategory pages, complete the following steps:

1. Double-click **catdisplay.jsp** to open the file in Page Designer.
2. Select **CATEGORY_NAME**, or the sample text you defined. The text displays with a reversed background, and blinks.
3. Press the Shift key while clicking the left mouse button on the thumbnail image. The text and image should now be selected, and should no longer be blinking.

Note

Selecting the elements on your page sometimes could be more difficult and sophisticated than you expect. If you choose the wrong elements before you insert the link, your JSP page will not work, and you may get an error from the JSP compiler.

4. From the menu bar, select **Insert -> Link**.
5. When the Attribute window appears, select the **Dynamic URL** tab, and enter the following:
 - URL: /webapp/commerce/servlet/CategoryDisplay
 - Parameters: click **Edit**.
6. When the URL Parameter Editor window appears, enter the following:
 - Name: merchant_rn (merchant reference number)
 - Select the **Specify by property** checkbox.
 - Click **Browse**.
7. When the Bean Property Selection window appears, select and expand the **childCategoryListBean**.
8. Select and expand the **List of child categories[]**. Select **Merchant reference number**, and then click **OK**.
9. Click **Add**.
10. When the URL Parameter Editor window appears, enter the following:
 - Name: cgrfnbr (category reference number)
 - Select the **Specify by property** checkbox, click **Browse**
11. When the Bean Property Selection window appears, select and expand the **childCategoryListBean**.
12. Select and expand the **List of child categories[]**. Select **Category reference number**, and then click **OK**.

13. Click **Add**.
14. Click **OK**, and then **OK** again.

Note

In this example, the URL passes parameters to the CategoryDisplay page.

15. After completing the previous steps, you may modify the page to give it a better look and feel. For example, the page could be modified as seen in Figure 72.

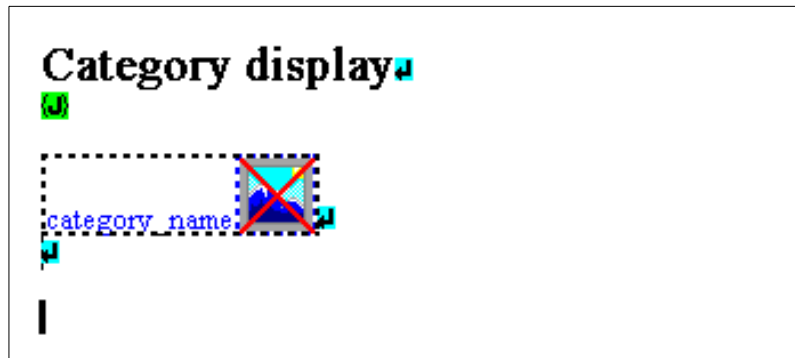


Figure 72. Category display page

16. From the menu, select **File** -> **Save**.
17. To further customize the category page template, use the features available in Page Designer or add beans of your choice.

Note

As a user browses through the categories, the lowest level of the category displayed should be the product list display. The category display command can reference a sub-category or a product list.

You can now test the JSP catalog page template, or integrate the JSP catalog page into a store. For testing refer to 15.3.4, "Testing JSPs" on page 356.

15.3.2 Creating a JSP product list page template

A product list page template is a type of category page template that lists the products within a category. Creating a JSP product list page template includes the following high level steps:

1. Create a JSP file.
2. Add ProductListBean to the JSP file.
3. Add dynamic product list information to the JSP file.
4. Create links to the product pages.

15.3.2.1 Create a JSP file

For detailed instructions for creating a file, refer to 15.3.1.1, “Create a JSP file” on page 337.

Note

Use the JSP file name productlistpage.jsp in place of catdisplay.jsp.

15.3.2.2 Add ProductListBean to the JSP file

For detailed instruction refer to 15.3.1.2, “Add the ChildCategoryListBean to the JSP file” on page 338.

Note

Use the bean name com.ibm.commerce.beans.ProductListBean in place of the com.ibm.commerce.beans.ChildCategoryListBean.

15.3.2.3 Add dynamic product list information to a JSP file

When creating a JSP product list page template, you can add dynamic product list information to the page by using the ProductListBean. The ProductListBean includes the property product[], which returns the list of products associated with the current category.

To add a dynamic product list information to a page, complete the following steps:

1. From the menu bar, select **Insert -> Dynamic Elements -> Dynamic Loop**.
2. When the Attribute window appears, click **Browse**.
3. When the Bean Property Selection window appears, select and expand the **productListBean** to display the properties.

4. Select **List of products[]**, and then click **OK**.
5. The `productListBean.product[]` is displayed in the Loop Property field in the Attribute dialog box. Click **OK**.
6. The Loop Property box should be displayed next to the {J} icon. Place the cursor in the Loop Property box.

Note

As you complete the following steps, ensure the cursor displays the hash mark, which denotes the loop property box.

7. Next we need to add the product short description, and thumbnail image.

Add the product short description

To add the product short description, complete the following steps:

1. From the menu bar, select **Insert -> Dynamic Elements -> Property Display**.
2. When the Attribute dialog box appears, click **Browse**.
3. When the Bean Property Selection dialog box appears, select and expand the **productListBean**.
4. Select and expand **List of products[]**. Select **Short description**, and then click **OK**.

The `productListBean.product[].shortDescription` should be displayed in the Property field in the Attribute window.

5. In the **Sample Text** field, type `PRODUCT_NAME`, or the sample text of your choice, and then click **OK**.

The sample text should be displayed in the products table. When the product list page is requested, the sample text will be replaced with information from the Commerce Suite database.

Add the thumbnail image

To add the thumbnail image, complete the following steps:

1. From the menu bar, select **Insert -> Dynamic Elements -> Dynamic Image**.
2. When the Attribute window appears, click **Browse** next to the SRC Attribute field.
3. When the Bean Property Selection window appears, select and expand the **productListBean**.

4. Select and expand **List of products[]**. Select **Thumbnail image path name**, and then click **OK**.
5. The `productListBean.product[].thumbnailPathName` is displayed in the SRC Attribute field in the Attribute window. Click **OK**.
The sample image should be displayed in the products table. When the product list page is requested, the sample image will be replaced with the image from the Commerce Suite database.
6. Add any other information you wish to your product list page.
7. If you are creating a product list template page, your next step is to create links to the product pages.

15.3.2.4 Create links to product pages

To create links to product pages, complete the following steps:

1. If not already open, double-click **productlistpage.jsp** to open the file in Page Designer.
2. Select **PRODUCT_NAME**, or the sample text you defined. Press and hold the Shift key, while clicking the left mouse button on the thumbnail image. The text no longer blinks.
3. From the menu bar, select **Insert -> Link**.
4. When the Attribute window appears, select the **Dynamic URL** tab, and enter the following:
 - URL: `/webapp/commerce/servlet/ProductDisplay`
5. Click **Edit** under the Parameters box to add the merchant reference number.
6. When the URL Parameter Editor window appears, enter the following:
 - Name: `merchant_rn` (merchant reference number)
 - Select **Specify by property** checkbox
 - Click **Browse**
7. When the Bean Property Selection window appears, select and expand the **productListBean**.
8. Select and expand the **List of products[]**, then select **Merchant reference number**. Click **OK**, and then click **Add**.
9. Click **Edit** under the Parameters box, to add the product reference number.
10. When the URL Parameter Editor window appears, click **Clear**, and then enter the following:

- Name: prrfnbr (product reference number)
 - Select **Specify by property** checkbox
 - Click **Browse**
11. When the Bean Property Selection window appears, select and expand the **productListBean**. Select **List of products[]**, select **Product reference number**, click **OK**, then **Add**.
 12. Click **OK**, then **OK** again.
 13. At this point you might want to add more elements and content to the page to improve the presentation quality (for example, refer to Figure 73).

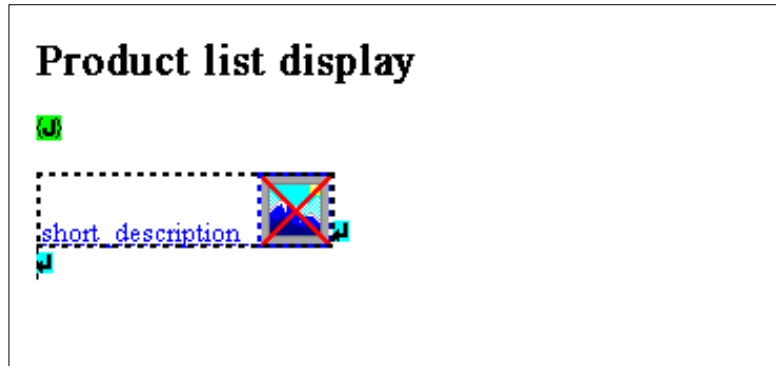


Figure 73. Product list display page

14. From the **File** menu, select **Save**.

At this point, you can test the JSP product list pages templates, or integrate the JSPs into the store, refer to 15.3.4, “Testing JSPs” on page 356.

15.3.3 Creating a JSP product page template

This section provides detailed instructions for creating a JSP product page template. Creating a JSP product page template includes the following high-level steps:

1. Create a JSP file.
2. Add the ProductBean to the JSP file.
3. Add dynamic product information to the JSP file.
4. Add product attributes and associated values (optional).
5. Make the attributes table dynamic (optional).

6. Further customize your product page, using the features available in Page Designer, or add more beans of your choice.
7. Add a product to the interest list.

15.3.3.1 Create a JSP file

For detailed instructions, refer to 15.3.1.1, “Create a JSP file” on page 337.

Note

Use the file name `productpage.jsp` instead of `catdisplay.jsp`.

15.3.3.2 Add a ProductBean to a JSP file

For detailed instructions, refer to 15.3.1.2, “Add the ChildCategoryListBean to the JSP file” on page 338.

Note

Use the bean name `com.ibm.commerce.beans.ProductBean`, instead of `com.ibm.commerce.beans.ChildCategoryListBean`.

15.3.3.3 Add dynamic product information to a JSP file

If you are creating a product page template, you will want to add product information to your page from the Commerce Suite database. Instead of adding static information, you can add information dynamically, using the `ProductBean`, which will be updated when your JSP file is requested from a browser.

To add dynamic product information, complete the following steps:

1. Position your cursor after the `{J}` icon.
2. From the menu bar, select **Insert -> Form and Input Fields -> Form**.

The Form and Input Fields toolbar should be displayed, and the form is added to your page. This form will allow your customers to add products to an interest list. You will complete the creation of this form later. Many product pages add dynamic product information for the following elements:

- Product description
- Product price and currency
- Product long description
- Product inventory

- Thumbnail image
- Position your cursor inside the form

Add dynamic product short description

To add a dynamic product short description, complete the following steps:

1. Place the cursor on the Page Designer form.
2. From the menu bar, select **Insert -> Dynamic Elements -> Property Display**.
3. When the Attribute window appears, click **Browse**.
4. When the Bean Property Selection window appears, select and expand the **productBean**.
5. Select **Short description**, and then click **OK**.

The productBean.shortDescription should appear in the Property field in the Attribute window.

6. In the Sample Text field, type `PRODUCT_NAME`, or sample text of your choice, and then click **OK**.

The sample text is displayed in the product page template. When the product page is requested, the sample text will be replaced with information from the Commerce Suite database.

Add product price and currency

1. From the menu bar, select **Insert -> Dynamic Elements -> Property Display**.
2. When the Attribute window appears, click **Browse**.
3. When the Bean Property Selection window appears, select and expand the **productBean**.
4. Select and expand **Price**. Select and expand **Currency display field**. Select **Currency display**, and then click **OK**.

The productBean.price.displayCurrency.presentationString should be displayed in the Property field in the Attribute window.

5. In the Sample Text field, type `PRICE` or sample text of your choice. Click **OK**.

When the product page is requested, the sample text will be replaced with information from the Commerce Suite database.

Add product long description

1. Repeat the steps listed in product short description above, selecting **Long description field 1** instead of short description. In the Sample Text field, type PRODUCT_DESC1 or sample text of your choice.
2. Repeat step 1 to select **Long description fields 2 and 3**.

Add product inventory

1. Repeat the steps listed in product short description above, selecting **Inventory** instead of short description. In the Sample Text field, type Inventory or sample text of your choice.

Add thumbnail image

1. From the menu bar, select **Insert -> Dynamic Elements -> Dynamic Image**.
2. When the Attribute window appears, click **Browse** under the SRC Attribute field.
3. When the Bean Property Selection window appears, select and expand the **productBean**.
4. Select **Thumbnail image path name**, and then click **OK**.
5. The productBean.thumbnailPathName should be displayed in the SRC Attribute field in the Attribute window. Click **OK**.

The sample image displays in the products table. When the product page is requested, the sample image will be replaced with the image from the Commerce Suite database.

6. To display product attributes and values, refer to the online help for adding product attributes and associated values.
7. To enable customers to add products displayed on this page to their interest list, or to order items, refer to the online help for adding a product to the interest list.

15.3.3.4 Add product attributes and associated values

Products in a catalog are often distinguished by a combination of distinct attributes and attribute values. To add product attributes and associated values to your product page, complete the following steps:

1. Double-click **productpage.jsp** to open the file in Page Designer.
2. Position the cursor in the form.
3. From the menu bar, select **Insert -> Table**.

4. When the Insert Table window appears, select the desired number of rows and columns. For this example, select one column and two rows, and then click **OK**.

Note

Inserting a table is optional. It is easier to find dynamic information within a table using Page Designer.

5. Select the **first row** within the table and enter the product attribute name by completing the following steps:
 - a. From the menu bar, select **Insert -> Dynamic Elements -> Property Display**.
 - b. When the Attribute window appears, click **Browse**.
 - c. When the Bean Property Selection window appears, select and expand the **productBean**.
 - d. Select and expand the **List of products attributes[]**. Select **Product attribute name**, and then click **OK**.

The productBean.productAttributeList[].attributeName is displayed in the Property field in the Attribute window.
 - e. When the Attribute window appears, enter the following:
 - Sample Text: ATTRIBUTE_NAME
 - Click **OK**

The sample text displays in the products table. When the product page is requested from a browser, the sample text will be replaced with information from the Commerce Suite database.
6. Select the **second row** within the table and enter the product attribute name by completing the following steps:
 - a. From the menu bar, select **Insert -> Dynamic Elements -> Property Display**.
 - b. When the Attribute window appears, click **Browse**.
 - c. When the Bean Property Selection window appears, select and expand the **productBean**.
 - d. Select and expand the **List of product attributes[]**, select and expand the **List of attribute values[]**. Select **Product attribute value**, and then click **OK**.

The `productBean.productAttributeList[].attributeName` is displayed in the Property field in the Attribute window

e. When the Attribute window appears, enter the following:

- Sample Text: **ATTRIBUTE_VALUE**
- Click **OK**

The sample text displays in the products table. When the product page is requested from a browser, the sample text will be replaced with information from the Commerce Suite database.

7. Repeat 15.3.3.4, “Add product attributes and associated values” on page 350 for additional products and associated values.
8. Next, we need to make the attributes table dynamic for the product attributes and associated values by completing the steps in the following section.

15.3.3.5 Make the attributes table dynamic

When you make the attributes table dynamic, each attribute loops through the attribute values. The list of attribute values make up the inner loop and the list of attributes makes up the outer loop.

1. Double-click **productpage.jsp** to open the file in Page Designer.
2. Double-click the Attributes table.

Note

Ensure that you click the table, not the cells.

3. When the Attribute window appears, select the **Dynamic** tab.
4. Select the **Loop** check box, and then click **Browse**.
5. When the Bean Property Selection window appears, select and expand the **productBean**.
6. Select and expand the **List of product attributes[]**, select **List of attribute values[]**, and then click **OK**.

The `productBean.productAttributeList[].productAttribute[]` is displayed in the Loop property field in the Attribute window.

7. In the Row/Column Range: Start field, type 2.
8. This row will be repeated for each value associated with the current attribute.
9. Select the **Outer Loop** check box, and then click **Browse**.

10. When the Bean Property Selection dialog window appears, select and expand **productBean**.
11. Select **List of product attributes[]**, and then click **OK**.
The `productBean.productAttributeList[]` is displayed in the Loop property field in the Attribute window.
12. When the Bean Property Selection window appears, click **OK**.
13. To verify the HTML source of your product in Page Designer, select the **HTML Source** tab to view the code generated by Page Designer.
If you are familiar with Java, you will see the two for loops in the generated code embedded in the HTML. The first loop iterates through the list of possible attributes. The second loop iterates through the list of associated values for a given attribute. If either of these lists are empty, the raised exception is caught within the generated code, and the body of the for loop with its embedded HTML tags will not be executed.
14. Select **File** -> **Save** to save your work.

Note

You can also loop through property lists by using the dynamic element Loop Property in the Page Designer. Loop Property allows you to repeat any series of HTML and JSP tags for each element in a list. Use Loop Property if you wish to repeat layouts other than table rows or columns.

15.3.3.6 Add a product to the interest list

In order to allow customers to add products to their interest list, you must complete the following steps:

- Complete the form for the `InterestItemAdd` command
- Add an interest list button
- Add an attributes radio button

Complete the form for the InterestItemAdd command

In this section, we will add attributes and values to the product page form for the `InterestItemAdd` command for the following attributes:

- `merchant_rn`
- `product_rn`
- `url`

To create a form for the InterestItemAdd command in your product page template, complete the following steps:

1. Double-click **productpage.jsp** to open the file in Page Designer.
2. Right-click the form, and select **Attributes**.
3. When the Attribute window appears, select the **Form** tab, and enter the following:
 - Action field: `/webapp/commerce/command/InterestItemAdd`
This is the C++ URL for the InterestItemAdd command
 - Method: Select the **Post** radio button
 - Select the **Hidden Fields** tab
4. From the Hidden Fields tab, enter the following:
 - Name: `merchant_rn`
 - Value: Select the **Specify by property** check box
 - Click **Browse**
5. When the Bean Property Selection window appears, select and expand **productBean**. Select **Merchant reference number**, and then click **OK**.
The `productBean.merchantReferenceNumber` is displayed in the Value field in the Attribute window.
6. Click **Add**.
7. From the Hidden Fields tab, enter the following:
 - Name: `product_rn`
 - Value: select the **Specify by property** check box
 - Click **Browse**
8. When the Bean Property Selection window appears, select and expand **productBean**, select **Product reference number**, and click **OK**.
The `productBean.productReferenceNumber` is displayed in the Value field in the Attribute window.
9. Click **Add**
10. From the Hidden Fields tab, enter the following:
 - Name: `url`
 - Value: `/webapp/commerce/command/InterestItemDisplay?merchant_rn=<%=java.net.URLEncoder.encode(productBean.getMerchantReferenceNumber())%>`
 - Click **Browse**

Note

Type the previous line carefully, because it is easy to mistype. Mistyping the code causes error on your page.

11. Click **Add**, then **OK**.
12. Your next step is to add an interest list button.

Add an interest list button

To add a button to your product page that allows customers to add products to their interest list, complete the following steps:

1. Double-click **productpage.jsp** to open the file in Page Designer.
2. Select the form in Page Designer.
3. Put your cursor at the bottom of the form, select **SUB** from the Form and Input Fields toolbar.
4. When the Attribute window appears, enter the following:
 - Name: InterestItemAdd
 - Label: Add to interest list
 - Ensure that **Submit** is selected as the Button Type
5. Add an attributes radio button to the form.
6. Select **File** -> **Save** to save your work.

Note

In this example the parameters are passed via a FORM, as fields of a form.

At this point you might want to add more content to the page. Your page should look similar to Figure 74.

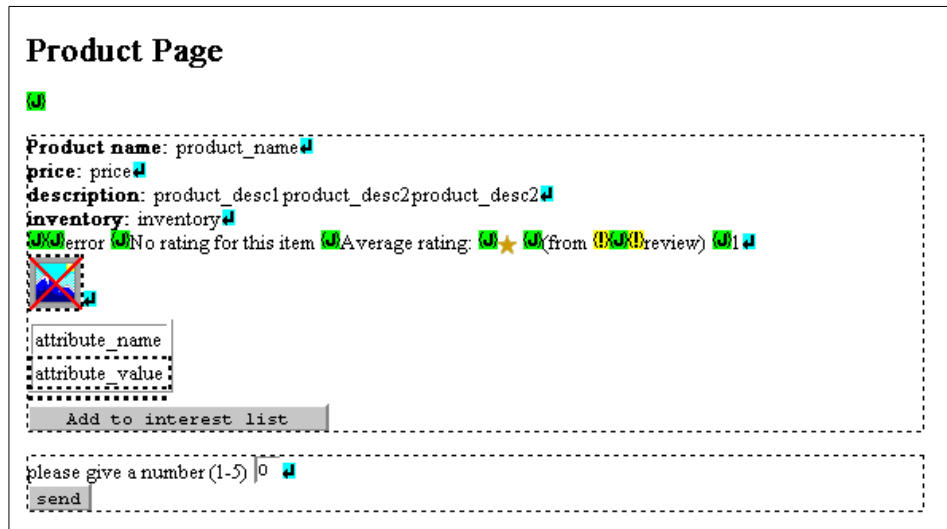


Figure 74. Product page

The product page template is now complete. At this point, you can test JSP catalog pages templates, or integrate JSP catalog pages into the store.

15.3.4 Testing JSPs

After you have created catalog page templates (category, product list, and product pages) using JSP technology, you can test that they produce the correct dynamic pages and that the pages are displayed properly.

To test your JSP catalog page templates, complete the following steps:

1. Publish the JSP catalog pages using WebSphere Commerce Studio. The files are automatically published to the correct location to be executed by the WebSphere Application Server and WebSphere Commerce Suite.
2. Open a browser and enter the appropriate URL:

- a. Category or product list pages URL:

<host>/webapp/commerce/<JSPfile>.jsp?merchant_rm=xxxx&cgrfnbr=yyyy

Where <host> is the hostname, <storeDir> is the store directory, <JSP file> is the name of the catalog page template to be tested, xxxx is the merchant reference number, and yyyy is the category reference number.

- b. Product pages URL:

<host>/webapp/commerce/<JSP file>.jsp?merchant_rm=xxxx&prfnbr=nnnn

Where `<host>` is the hostname, `<storeDir>` is the store directory, `<JSP file>` is the name of the catalog page template to be tested, `xxxx` is the merchant reference number, and `yyyy` is the product reference number.

Note

You can get the `merchant_rn`, `cgrfnbr`, and `prfnbr` from the WebSphere Commerce Suite Administer by browsing the categories.

Also, you can find them applied to the URL as a parameter if you step into the `demomall` (or any existing store) and check a category or product page with the source view under your browser.

The appropriate catalog page should be displayed. If the appropriate catalog page does not display, or an error page is displayed, see the 15.3.6, “JSP development recommendations” on page 361.

3. Review the page.

If the catalog page you requested does not display, or if an error page is displayed, complete the following:

- a. Confirm that you entered the correct URL, including correct name, number, and value.
- b. Ensure that the catalog page was published to the correct location on the Commerce Suite server:

Windows NT: `<drive:><storeDir>`

AIX: `/<storeDir>`

- c. View the HTML source for the JSP file. Ensure that the necessary looping properties are established for category and product list pages, or for attributes and attribute values for product pages.
- d. Change the trace level, and use the logs to debug the problem.
- e. If the following error `java.lang.ClassFormatError: Wrong name displays`, stop and start the WebSphere Commerce Server in the WebSphere Application Server Administrator’s Console.

15.3.5 Deploying JSPs

Now that you have created the catalog pages (category, product list, and product pages) using JSP and bean technology, you can integrate the JSPs into your store.

Stores created with Store Creator or stores that include the default `Net.Data` macros use the `C++ CatalogDisplay` and `ProductDisplay` commands to

display the existing catalog pages created with Net.Data macros. To integrate your JSP catalog pages in the store, you must replace the C++ CatalogDisplay and ProductDisplay commands with the corresponding Java commands.

15.3.5.1 Deployment prerequisites

The following instructions for integrating the catalog make the following assumptions:

- Your store was created using Store Creator, and included the sample products in that store.
- You created the following JSP catalog pages: category, product list, and product. For more information, see the Create a category page template, the Create a product list page template, and the Create a product page template.
- the JSP catalog pages are in the root of the <storedir> folder in the project directory for your store in WebSphere Commerce Studio. For more information, see the WebSphere Commerce Suite store file structure.

15.3.5.2 Test the JSP

To integrate the JSP catalog pages into a store, complete the following steps:

1. Test the JSP catalog pages.
2. (*Optional*) Delete the following files in the project directory for your store:

<storedir>\macros\catdisp.d2w
<storedir>\macros\proddisp.d2w

These are the original Net.Data category and product page templates for the store, created by the Store Creator, and will be replaced by the JSP catalog pages.

3. The following macros, located in <storedir>\macros in the project directory for your store in WebSphere Commerce Studio, contain C++ CatalogDisplay and ProductDisplay commands:

- nav.d2w and/or navbottom.d2w
- searchrs1t.d2w
- regNew.d2w
- regUpdate.d2w
- shopcart.d2w

In the above macros and any others you may have added, replace all instances of the C++ `CatalogDisplay` and `ProductDisplay` commands with Java commands, by doing the following:

- a. Open the files, mentioned before, one-by-one, and follow the steps.
- b. Search for the **CategoryDisplay** word.
- c. Replace the path:
from: `/webapp/commerce/command/CatalogDisplay`
to: `/webapp/commerce/servlet/CatalogDisplay`
- d. Ensure that you have the right parameters in your command, the `merchant_rn`, and also the `cgrfnbr` parameters, the command should look like this: `CategoryDisplay?merchant_rn=xxxx&cgrfnbr=yyyy` (where `xxxx` is the merchant reference number, and `yyyy` is the category reference number).
- e. Search for the **ProductDisplay** word.
- f. If you find, then replace the path `/webapp/commerce/command/ProductDisplay` with `/webapp/commerce/servlet/ProductDisplay`
- g. Ensure that you have the right parameters in your command. The `merchant_rn`, and also the `prrfnbr` parameters should look like the following:
`ProductDisplay?merchant_m=xxxx&prrfnbr=yyyy`
Where `xxxx` is the merchant reference number, and `yyyy` is the product reference number.

Now you have to change the template files assigned to the categories, and the products. There are two possible method for that:

- Doing one-by-one under the WebSphere Commerce Suite Administrator. This could take very long time, depending on your products and categories, but you can even set different pages even for each category or product.
- Modifying the mass import input file, and importing the file again.

Important

Do not forget if you decided to use JSP, you must use them for all category page, and all product page.

15.3.5.3 Category template assignment using WCS Administrator

For detailed instructions for updating templates using WebSphere Commerce Suite Administrator refer to the *IBM WebSphere Commerce Suite Pro Edition, Commerce Suite Administrator V4.1* product guide.

15.3.5.4 Category template assignment - mass import

In the mass import XML file you have different sections for your store information.

- Category template definition
- Product list template definition
- Product template definition

Category template definition

The category template definitions can be found within the following tags:

```
<catesgp cscgnbr_cgnbr="nnnn" csdisplay="tttt"/>
```

Where the nnnn is the category name (for example "CD"), and the tttt is the template URL (for example "/catdisp.jsp").

Product list template definition

Under the category template definitions you can find those categories, which are at the lowest level. To find them check the following tags:

```
<cgprre1 cpcgnbr_cgnbr="cccc" cprnbr_prnbr="pppp"/>
```

In this tags you can define which product belongs to which category, where cccc is the category name (for example "CD"), and pppp is the product name (for example "CD-00002"). Find all the different categories under these tags, those are at the lowest category level. Then refer to the previous description, where to find and modify your category display templates; and despite of using the category display page URL, use the product list display URL (for example "productlistpage.jsp").

Product template definition

The product template definitions can be found within the following tags:

```
<prodsgp psprnbr_prnbr="pppp" psdisplay="tttt" psdesc="dddd"/>
```

Where pppp is the product number (for example "CD-00002"), and tttt is the template URL (for example "/productpage.jsp"), and dddd is the description of the template (for example "Product Template").

You only have to replace the template URLs with your JSP pages URL. Use your editor to replace them with the required template. For example just make

a search and replace for both of templates; search for the old template file, and replace with the new one.

Note

First, you should try the JSP page URL, before you replace the templates! Be careful with the URL, it could be relative (for example “catdisp.jsp”), or absolute (for example “/catdisp.jsp”). It is recommended that you use absolute URL assignment.

For more information on mass import refer to 13.3.1, “Add Category and Product data - massimport XML” on page 286.

While testing your store, you may notice that the style of the catalog pages does not match the style of the rest of your store pages. To change the style of the catalog pages, in order to change the look and feel of your pages refer to the product documentation *Creating a Store Version 4.1*, found on the product CD (file name createstore.pdf).

15.3.6 JSP development recommendations

Sometimes the WebSphere Application Server does not recognize your JSPs. In order to manage this problem you may do the following:

- Under the WebSphere Application Server Administrator’s Console, select the **Topology** tab, then follow the structure: **<your host>** -> **WebSphere Commerce Server** -> **WCS Servlet Engine** -> **WCSServlets**. Right-click the **WCSServlets** label and select the **Restart Web App** menu, which restarts your servlet engine.

Note

If you deploy a new servlet you have to restart your servlet engine. Use the process described before.

- There is a more efficient method restarting your application server: under the **Topology** tab, **STOP** and **START** the WebSphere Commerce Server.
- JSPs are compiled into servlets, these servlets are temporary class files, stored in the following directory:

<WebSphere Commerce Suite dir>\WAServer\temp\default_host\WCSServlets

You can delete them in order to get rid of the old JSPs.

Note

Once your servlets and JSPs have started they stay in the memory, sometimes deleting them from the directory mentioned previously, does not help. In this case you have to restart your servlet engine.

- In order to get around with caching, you should set your Web browser cache to 0 memory cache and also 0 disk cache, and force the browser to reload the page (use Shift-reload).
- You should use versioning in your JSP pages, just put a vX.X text at the end of your JSP, or put it into a comment, then you are sure about which is the current servlet you are testing.
- Try to use unique name for your JSPs, do not use CategoryDisplay.jsp, because it is more difficult to trace back your requests.
- If you see the following Java error lines during the test:

Unhandled error! You might want to consider having an error page to report such errors more gracefully

```
javax.servlet.ServletException at  
com.ibm.commerce.beans.DataBeanManager.activate(DataBeanManager.java:70)  
..  
.
```

This means that you have the wrong parameters in your URL, for example the merchant_rn=xxxx (where xxxx is the merchant reference number) is missing.

15.4 Custom servlet and bean example

This example explains how to create custom servlets and beans to perform a product review for a store. In this example, the customer can rate the products from 1 to 5.

The following fields are found on the product display page:

- Review - rates the book on a scale of 1 to 5
- Review text - reviewers comments about the product

After the user submits a review, they are sent a confirmation number to acknowledge that the product review transaction was successful. From the review acknowledge page the user can step back to the product display page.

Each review is stored within the PRODUCT table in the Commerce Database. There are six customizable fields provided within the PRODUCT table. In this example, the review fields use two of the customizable fields, as displayed in Table 33.

Table 33. Product customizable fields used in the example

Example fields	PRODUCT table customizable fields
review	PRFIELD1
review text	PRFIELD2

- Review - PRFIELD2, and one for the sum of the given numbers (PRFIELD1). The calculation is very simple: stars = (sum of the numbers) / (number of reviews).

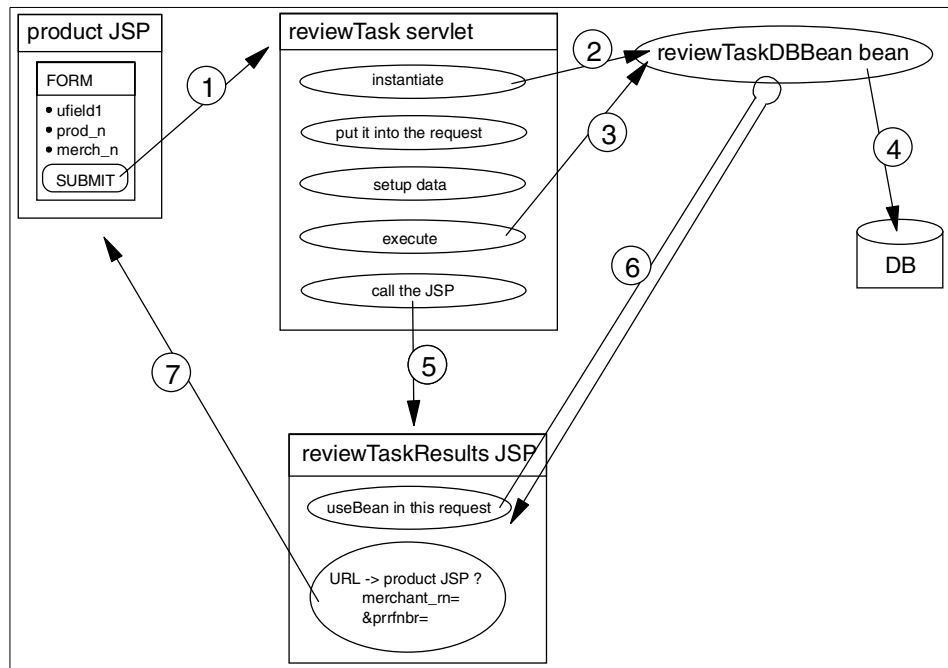


Figure 75. The review example

The following developments and modifications necessary on the store, step-by-step:

1. Create the database query with the SQL Wizard.
2. Create the JavaBean and the servlet with the Database wizard.

3. Modify the generated elements (bean, servlet).
4. Modify the existing Product Display JSP.

15.4.1 Creating a database query

In our example the business logic is represented by a database update, in case you have to write your own database command.

To create your SQL in WebSphere Commerce Studio, complete the following steps:

1. On the left panel click on <storeid> folder, where <storeid> is the name of your store.
2. From the menu bar, select **Tools -> Wizards -> SQL Wizard**.
3. The SQL Wizard window appears, fill the text box with your SQL query name, type in review, and click **Next**.

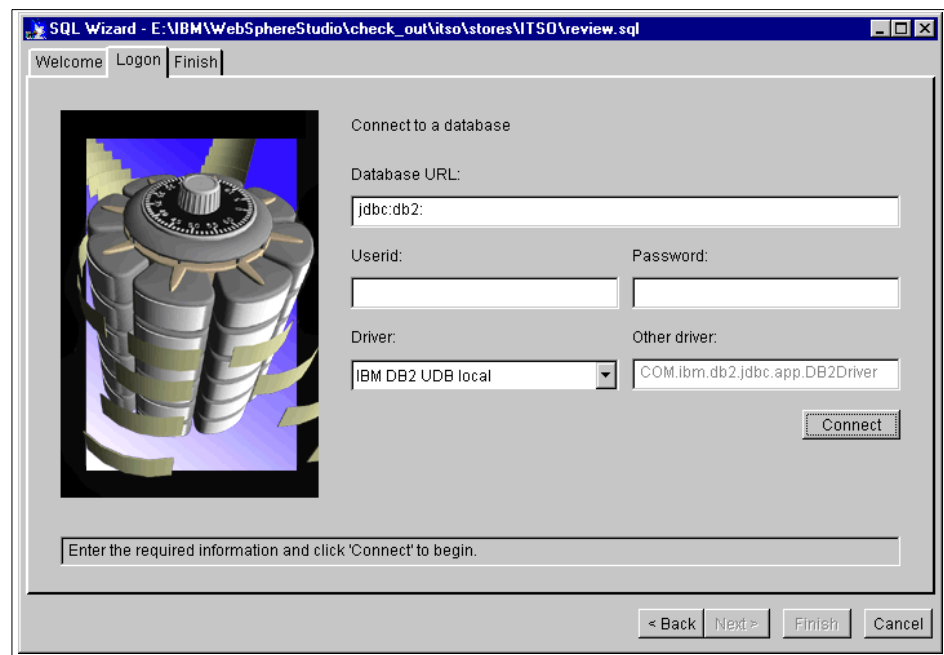


Figure 76. SQL Wizard - Logon tab

4. The tab displayed in Figure 76 is for the database connection. Enter the following:

- a. Fill the database URL, jdbc:db2:<your database>, in our example the database name is: dbwcsnt.
 - b. Enter the user ID, and the password for the database connection.
 - c. Choose your driver.
 - d. Click **Connect**.
5. After your connection is successful, continue to the tables.
 6. Select the SQL command and table.
 - a. Select the SQL command, select the **Update** radio button.
 - b. Select the required table for your SQL, select <USERID>.PRODUCT., the small checkmark appears left of the table name.
 - c. Click **Next**.
 7. Here you can choose your columns you want to be updated.
 - a. Click the Value field, beside the **PRFIELD1** column, and click **Parameter...**
 - b. Enter a name for your parameter, type ufiled1, then click **OK**.
 - c. Click **Next**.
 8. On the next tab you find the conditions to select your rows in the table.
 - a. Select **PRRFNBR** column (stands for the product reference number), and then select **is exactly equal to**.
 - b. Click the first line under the Values, and then click **Parameter...**
 - c. Enter the name of your parameter, type prod_n, and click **OK**.
 - d. Click **Find on another column**, another condition tab appears.
 - e. Select **PRMENBR** column (stands for the merchant reference number), select the **is exactly equal to** operator.
 - f. Click the first line under the Values, then click **Parameter...**
 - g. Enter the name of your parameter, type merch_n, and click **OK**.
 - h. Click **Next**.
 9. The next tab describes your SQL statement.

Optional

You can try your SQL statement clicking on **Run SQL...**, in this case you will need correct values from your store.

10. To finish the SQL wizard, click **Finish**.

11. The review.sql appears under the <storedir> folder.

You are ready to use this SQL statement under the database wizard, or in any other context.

15.4.2 Creating a bean and servlet

The business logic realization for this example is a bean that accesses the database table, and makes the necessary modifications.

To create a bean and servlet under WebSphere Commerce Studio, complete the following steps:

1. On the left panel click on the <storedir> folder.
2. From the menu bar, select **Tools -> Wizards -> Database Wizard**.
3. When the Database Wizard window appears, the message select your SQL statement you want to use? is displayed, select **review.sql** from the drop-down box, and then click **Next**.
4. In the following window you decide which kind of page you want to generate.
 - Create an input page
 - Create a result page
 - Click **Next**.
5. In the following window you can select the required fields you want to specify on your input page. Check the following:
 - ufield1
 - prod_n
 - merch_n
 - Click **Next**.
6. Similar to the previous window, select the fields you want to display on the result page. Check on the followings:
 - merch_n
 - prod_n
 - ufield1
 - Click **Next**.
7. In the Methods window you can specify which JavaBeans methods you want to run. Leave the execute() method checked. Click **Next**.

8. When the message Will you use this bean or query on more than one page? is displayed, select **No**.

Note

Here you need to decide the scope of your bean instance. If you want to use the bean for the whole session, select Yes. If you want to use the bean just for the request, select No.

9. Select **No**. Specify your bean name **reviewBean**, and then click **Next**.
10. In the Finish window you can find all the files that will be generated, and you can rename them. Click **Rename**, leave the package name, and type in under the prefix: reviewTask. Click **OK**, then click **Finish**.
11. The following files are generated:
- reviewTaskInput.html
 - reviewTaskResults.jsp
 - reviewTaskDBBean.java
 - reviewTaskDBBean.class
 - reviewTask.java
 - reviewTask.class
 - reviewTask.servlet
12. The skeleton is now ready for any needed modifications.

15.4.3 Modify the existing product display JSP

To modify the existing product display JSP, complete the following tasks:

1. Modify the servlet configuration file.

In order to access the JSP, we have to set the right URI for the JSP and also modify the datasource name.

- a. From the right-hand panel (make sure you are using the Publishing View) drag the **reviewTask.servlet** and drop it onto the servlet folder (one folder above the original position).

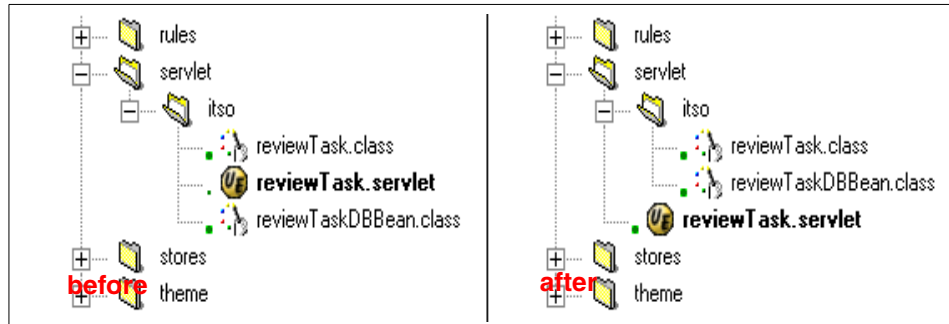


Figure 77. Replace the *reviewTask.servlet*

- b. Double-click the **reviewTask.servlet** file. The file appears in the editor (Notepad by default). This is an XML configuration file for your servlet. It contains the information about the database access, and the JSP URL which the servlet calls. Here you can modify the user ID, and the password anytime.
 - c. Change the `<page-list><default-page><uri>` field to the correct URI for the `reviewTaskResult.jsp`, by typing the following:


```
/servlet/<your store>/reviewTaskResults.jsp
```
 - d. To change the datasource name for the servlet, find the line:


```
<init-parameter value="jdbc/jdbcdb2dbwcsaix" name="dataSourceName"/>
```

 Change it to the following:


```
<init-parameter value="jdbc/CommerceDataSource" name="dataSourceName"/>
```

 The Datasource Name parameter can be found in the WebSphere Application Server Administrator's Console, under the Topology tab, CommerceDataSource node name.
 - e. Save the file and close the editor.
2. Modify the bean.

The SQL command as designed needs some modification to fit our expectations related to the business process. The number of reviews should be increased by 1, and the number of stars given should be increased by the given value.

 - a. From the left-hand panel, double-click **reviewTaskDBBean.java**.
 - b. When the Check-out files with dependencies window appears, check in the dependent files, and click **Check out and Edit**.
 - c. In the Java file find the following variable:

protected java.lang.String SQLString=

- d. Change the PRFIELD1 = ? to the following:

```
PRFIELD1 = ( PRFIELD1 + ? )
```

- e. Add the following line after the previous line:

```
, PRFIELD2 = ( PRFIELD2 + 1 )
```

Note

Do not forget the comma before the expression.

- f. Your new SQL statement should look like this:

```
UPDATE COMMERCE.PRODUCT SET PRFIELD1 = (PRFIELD1 + ?) , PRFIELD2 = ( PRFIELD2  
+ 1 ) WHERE ( ( <USERID>.PRODUCT.PRRFNBR = ? ) AND ( <USERID>.PRODUCT.PRMENBR  
= ? ) )
```

- g. Save the file and close the editor.
- h. From the left-hand panel, right-click the **reviewTaskDBBean.java** file. Click **Compile**, and then click **Yes To All**.

3. Modify the input page.

The servlet URL does not fit into our structure.

- a. Double-click the file **reviewTaskInput.html**. The file will be opened in Page Designer.
- b. When the Page Designer appears, double-click the form frame (the biggest frame on the page).
- c. When the Attributes window appears, change the Action text-box to:
`/webapp/commerce/reviewTask`

Note

This is the URI for the reviewTask servlet.

- d. Save the file and close the editor.

4. Modify the result page.

In order to get back to the Input page, you have to create a link to the requested URL.

- a. Double-click the file **reviewTaskResults.jsp**. The file will be opened in Page Designer.

- b. At the end of the page type your text:
Thanks for your review
go back
- c. To create a link back to the input page, select **go back**, and then select **Insert -> Link** from the menu bar.
- d. When the Link Attributes window appears, do the following:
 - File Name: reviewTaskInput.html
 - Click **OK**.
- e. Save the file and close the editor.

15.4.4 Publish the files under WebSphere Commerce Studio

In order to publish your new files, do the following:

1. Check-in all your files, select all of the checked-out files (using the Ctrl button on your keyboard), right-click one of the selected files and click **Check-in**.
2. Make sure you are in the publishing view, and right-click on <your server> node http://<your server>. Right-click and select **Properties -> Define Publishing Targets**.
3. Define the target paths for your files.
4. Select the **servlet** row, and change the directory to the current servlet directory (for example: d:/projects/itso/servlet).
5. Click **OK**, and **OK** again.
6. Right-click <your server> node (http://localhost).
7. Select **Publish This Server**, and then click **OK**.
8. The publishing starts...

15.4.5 Deploy the servlet under WebSphere Commerce Suite

Before you can use your new servlet, you have to deploy under the WebSphere Commerce Server.

1. Start WebSphere Administrator's Console, by clicking **Start -> Programs -> IBM WebSphere -> Application Server v 3.0 -> Administrator's Console**.
2. Under the Tasks tab, open **Configuration** (Click +), then click **Add a servlet**.
3. Click the **Green lamp**, to start the task.

4. From the right-hand panel, select **No**, and then click **Next**.
5. Select the servlet container by clicking **WebSphere Administrative Domain -> Nodes -> <your host> -> WebSphere Commerce Server -> WCS Servlet Engine -> WCSServlets**, and click **Next**.
6. Select **Create User Defined Servlet**, and then click **Next**.
7. When the Create User Defined Servlet panel appears, as seen in Figure 78, enter the following:
 - Servlet Name: reviewTask
 - Web Application: WCSServlets
 - Servlet class name: <your store>.reviewTask
 - Click **Add**
 - Enter the following: /webapp/commerce/reviewTask
 - Click **OK**.

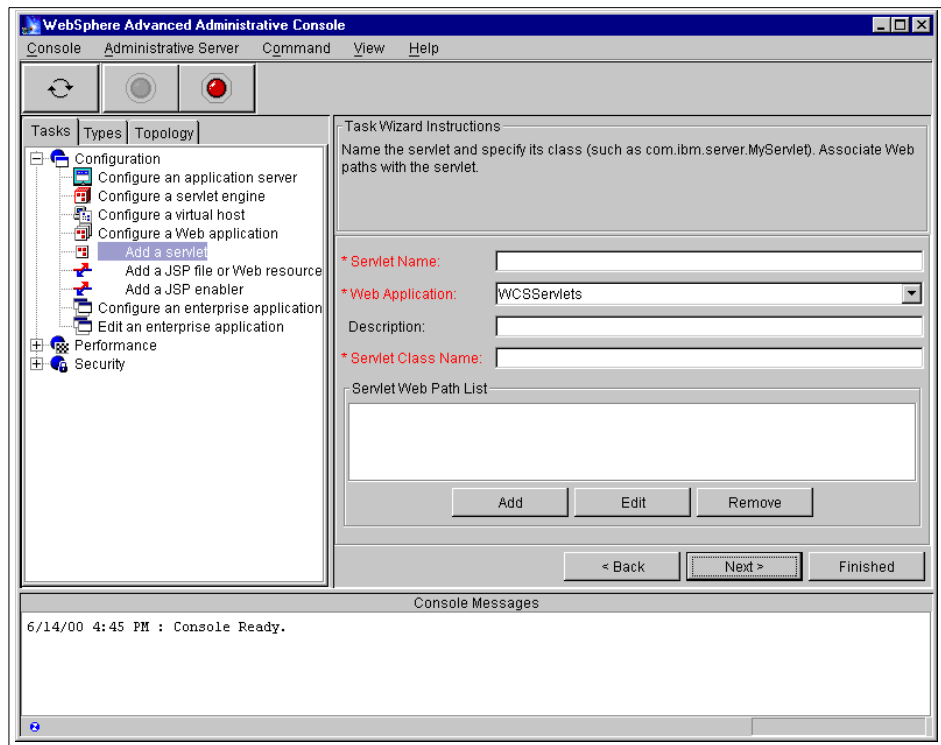


Figure 78. Administrator's Console, Add a servlet

8. Click **Finished**, wait for the window to appear, and click **OK**.
9. Also you have to add your servlet path to the classpath. Under the topology tab select the **WebSphere Administrative Domain -> Nodes -> <your host> -> WebSphere Commerce Server -> WCS Servlet Engine -> WCSServlets** line.
10. From the right-hand panel select the **Advanced** tab.
11. On the right-hand side of the upper table, click the ... button.
12. When the directory selector window appears, select your servlet directory (for example: /projects/stores/ITSO/servlet), then click **Open**.
13. Click **Apply**, at the bottom of the right-hand panel.
14. Wait until the progress bar finishes the process, and in the log window appears with the message ...completed successfully.
15. Restart your Web application server.
 - a. Select the **WebSphere Administrative Domain -> Nodes -> <your host> -> WebSphere Commerce Server** line, and then click the **Red lamp**.
 - b. Wait until the window appears, and then click **OK**.
 - c. Click on the **Green lamp**.
 - d. Wait until the window appears, and then click **OK**.
16. Your Application Server is ready.

15.4.6 Testing the new servlet and bean

Before you step forward to integrate your new servlet and bean into the store, you should make sure that everything is correct. Do the following to test your application:

1. Open your browser, and type the following URL:
`http://<your_host>/<your_store>/reviewTaskInput.html`
2. Fill the fields with valid data, enter 5 in the ufield1 to make it smaller.
The prod_n field represents the product reference number. The merch_n represents the merchant reference number.

Note

You need to use valid data from your Commerce Suite database. Follow the steps to acquire the numbers in the note at 15.3.4, "Testing JSPs" on page 356.

3. Click **Submit**.
4. The resulting page should return with the correct values.

15.4.7 Integrating into the store

To integrate the existing reviewTask Web application module into your store, complete the following steps:

- Modify the Product Display JSP.
- Modify the review acknowledgment page.

15.4.7.1 Modifying the Product Display JSP

First you might want to show the actual review status of the product. Modify the existing productpage.jsp file to access the review information, and put into the page. Select the **HTML Source** at the bottom of your Page Designer.

1. The following code could be placed anywhere in the product.jsp In this example the stars appear after the product description and before the product picture.

- a. Original JSP section:

```
<WSPX:PROPERTY property="productBean.inventory" authorimetext="inventory">
--><%= productBean.getInventory() %><!--METADATA type="DynamicData"
endspan--><BR>
```

- b. The New JSP section:

```
Average rating:
<%=! int allstar,votes,stars; %>
<%=
try {
    allstar=(int)Integer.parseInt(productBean.getField1());
    votes=(int)Integer.parseInt(productBean.getField2());
} catch (NumberFormatException e) {
%>
error
<%=
}
if(votes==0) {
%>
No rating for this item
<%=
} else {
    stars=allstar/votes;
%>
Average rating:
```

```

<%
    for (int c=0 ; c<stars; c++) {
    %>
<IMG src="/<storedir>images/star.gif">
    <%
        }
    %>
    (from <B><!--METADATA type="DynamicData" startspan<WSPX:PROPERTY
property="productBean.field2">-->
<%= productBean.getField2() %>
<!--METADATA type="DynamicData" endspan--></B> review)
    <%
        }
    %>
<BR>
Here you have to define the original path for the image (star.gif). You can
hardcode in the file, or even you can add your image visually.
<IMG src="star.gif">
    <%
        }
    %>(from <B><!--METADATA type="DynamicData" startspan
<WSPX:PROPERTY property="productBean.field2">
--><%= productBean.getField2() %><!--METADATA type="DynamicData"
endspan--></B> review) <BR>
original JSP section starting:
<!--METADATA type="DynamicData" startspan
<IMG dynamicElement border="0" property="productBean.thumbnailPathName">

```

- c. Save your file by clicking **File** -> **Save**. Then select the Normal view, at the bottom of Page Designer.
2. The next step is to place the voting form at the bottom of the page. You can copy and paste HTML code from the reviewTaskInput.html, or just use it to make sure your code is correct.
3. Open **productpage.jsp** in the Page Designer.
4. Create a form at the bottom of the page (be careful not to click inside the existing form). From the menu select **Insert** -> **Form and Input Fields** -> **Form**.
5. Double-click the new form box.
6. When the Form Attributes window appears, do the following:
 - Action: /webapp/commerce/reviewTask
 - Set the Method to **Post**.

- Set the **Target** to **Same Frame**.
7. Click the **Hidden Fields** tab.
 8. In the Name field type: prod_n.
 9. Select the **Value** field, and check the **Specify by property** check-box.
 10. Click **Browse**, and select the **productBean -> product reference number** property. Click **OK**.
 11. Click **Add**. Now you have added your hidden field.
 12. Add the next field by following the previous steps:
 - Name: merch_n
 - Value: Click **Specify by property -> productBean -> merchant reference number**.
 13. Click **Extended...**, and select the **Event** tab.
 14. Select the **OnSubmit** event, and then click **Add**.
 15. Click the **OnSubmit** line under the Event, in the Action table.
 16. Click the Script text-field, and type the following validation script:

```
if(this.ffield1.value<1 || this.ffield1.value>5) { alert('wrong number'); return false; } return true;
```
 17. Click **Configure**.
 18. Click **OK**, then **OK** again.
 19. Click in the form, and type the following:

Send your review (give a number from 1-5)
 20. Insert the text-box, **Insert -> Form and Input Fields -> Text Field**.
 21. When the Text Field attributes window appears, do the following:
 - Name: ufield1
 - Columns: 2
 - Maximum Length: 1
 - Initial value 0
 - Click **OK**.
 22. Insert the Submit button by selecting **Insert -> Form and Input Fields -> Push Button -> Submit Button** from the menu bar.
 23. When the Attributes window appears, enter the following:
 - Name: reviewSubmit

- Label field type: send

24. Your product page should look similar to Figure 79.

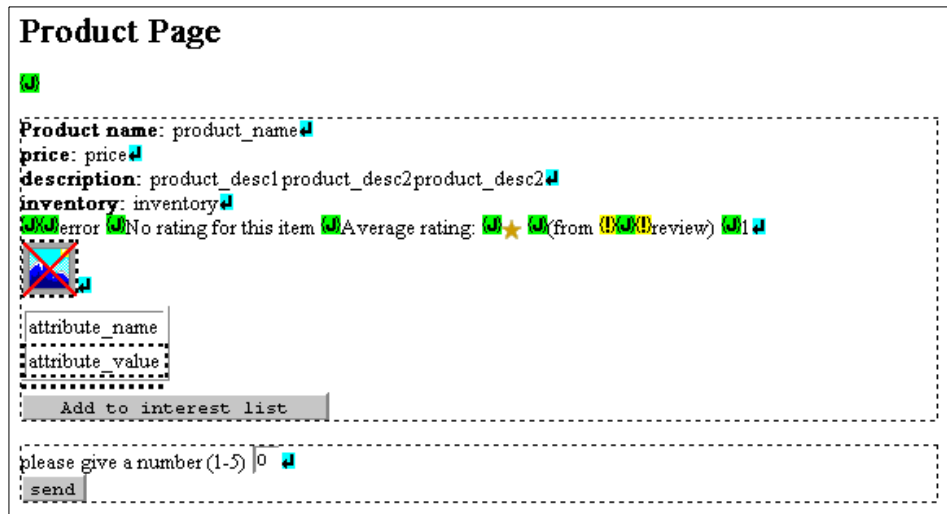


Figure 79. Modified product page

15.4.7.2 Modify the review acknowledgement

After a user sends the product review, a review acknowledgment should be sent to the user. Knowing that the transaction was successful will reduce the possibility that users will enter the same review many times.

25. Open the **reviewTaskResults.jsp** in the Page Designer.

26. To delete the table from the page, click the table (select the whole table), and then select **Table -> Delete table** from the menu bar. Two {J} tags will remain.

27. Select the text and link at the bottom, and **cut** (Ctrl-X), then **paste** (Ctrl-V) between the two green {J} tags. The page should look similar to Figure 80.

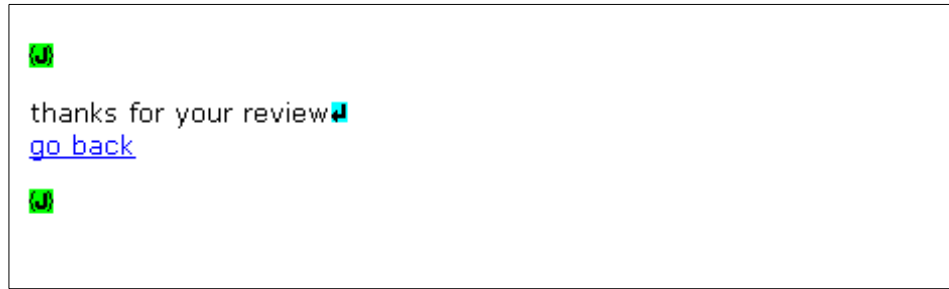


Figure 80. Page Designer - reviewTaskResults.jsp visual design

28. Right-click **link** (go back text), and then select **Attributes**.
29. When the link Attributes window appears, select the **Dynamic URL** tab, and type the following in the URL:
`/webapp/commerce/servlet/<your store>/<your product page>.jsp`
30. Under Parameters, click **Edit**, and enter the following:
 - Name: prrfnbr
 - Value: **Specify by property**
 - Select: **reviewBean>prod_n** and click **Add**.
 - Name: merchant_rn
 - Value: **Specify by property**
 - Select: **reviewBean -> merch_n** and click **Add**.
31. Click **OK**, and **OK** again.
32. Save the file, and close the editor.
33. Publish your store with all modifications.

15.4.8 Testing pages under the store

Open the store, click the catalog link, and drill down until the product appears. If you have any errors, check your JSPs, restart your WebSphere Application Server, or restart your WebSphere Commerce Suite server.

15.5 Where to find more information

IBM Publications

- *Developing an e-business Application for the IBM WebSphere Application Server*, SG24-5423
- *Patterns for e-business: User-to-Business Patterns for Topology 1 and 2 using WebSphere Advanced Edition*, SG24-5864
- *IBM WebSphere Commerce Suite Pro Edition, Commands, Task, Overridable Functions and Database Tables V4.1*
- *Servlet and JSP Programming with IBM WebSphere Studio and VisualAge for Java*, SG24-5755
- *Design and Implement Servlets, JSPs, and EJBs for IBM WebSphere Application Server*, SG24-5754

Chapter 16. Personalization in WebSphere Commerce Suite

Implementing a strong personalization strategy is a way to develop customer loyalty and complement marketing campaigns for your commerce site. Many commerce sites are making use of some form of personalization strategy by customizing Web pages to the interests and needs of each visitor. Some of the most common forms of personalization are as follows:

- Merchandising - cross-sell, up-sell, accessories, and substitution.
- Inventory- recommend overstocked items.
- Legal Policy - allow or disallow the sale of products based upon legal policies.
- Time based - seasonal, event, holiday, daily or monthly sale.
- Affiliate program - recommend items in the catalog to particular customers.

This chapter is organized into the following sections:

- Personalization overview
- Personalization example
- Personalization debug tips

16.1 Personalization overview

The WebSphere Commerce Suite provides a rules-based solution for generating personalized product recommendations to designated customers. The personalization software has two components:

- Blaze Advisor Builder

The Blaze Advisor Builder is included in WebSphere Commerce Studio, Professional Developer Edition, and is used to create the rule project.

- Blaze Advisor Server

The Blaze Advisor Server is included in the WebSphere Commerce Suite, Pro Edition, and is used on the WebSphere Commerce Server. The advisor server accepts incoming information from the shopping session to evaluate the rules, and then determine a list of products to recommend to the shopper. The list of recommended products are displayed in dynamic HTML pages generated by net.data macros.

16.1.1 Blaze Advisor Builder

WebSphere Commerce Studio, Professional Developer Edition includes the Blaze Advisor Builder, which serves as the development environment for rule creation. The Advisor Builder includes both graphical and textual editors. Use the Advisor Builder to create a collection of rules, known as a rule project.

A rule project is a set of files that contains all of the information that your WCS site requires to interact with the rule engine. It is the highest level container for all of the rule-related information for your personalization application. There can be only one rule project per WebSphere Commerce Suite site whether the site is a single store or a mall containing many shops.

16.1.1.1 Considerations before creating a rule project

Prior to creating a rule project, you should have a thorough knowledge of its design rationale. There are many ways that personalization strategies can be used to implement marketing campaigns. Figure 81 provides a summary of the personalization strategies integrated with personalization within the WebSphere Commerce Suite.

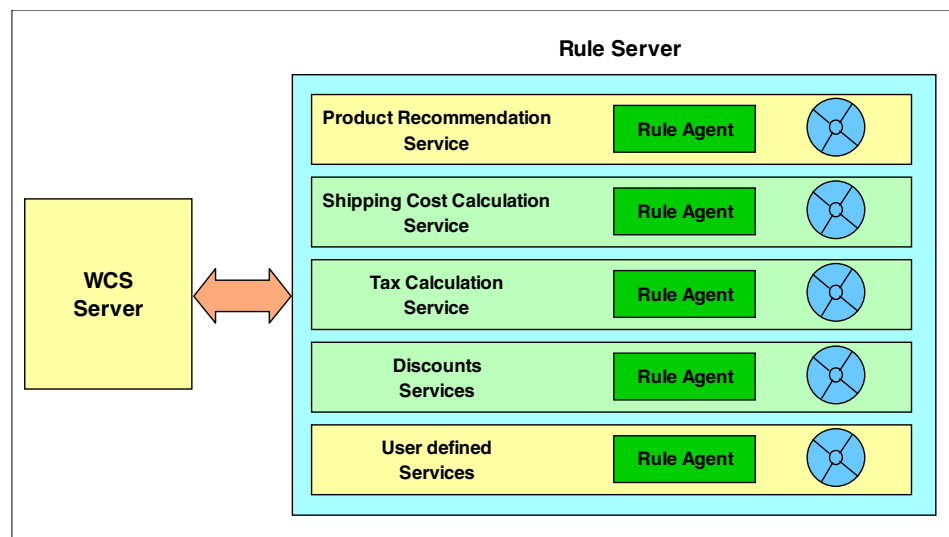


Figure 81. Potential usage of rule-services

It is important to note that when we create a Blaze Advisor Rule project in WebSphere Commerce Studio, it is advisable to select a master rule project. By default, there is one master rule project called RecommendationMaster. Based on it, you can create a rule project that is capable of providing a personalized product recommendation to customers.

There are a number of issues that you should be aware of before you start:

The Blaze Advisor rule server creates an instance of the Userclass as an object named `currentUse` when being called. This is the name of the object for the entire rule processing session. Use this name in your rules when referring to a particular member of the user class.

- Only one shopping cart can be the current shopping cart in the rule project although you may have multiple shopping carts in your site. The Blaze Advisor rule server concatenates all current `interestItemLists` in the database to create a single `current interestItemList` in the rule project.
- The Blaze Advisor Builder is case-sensitive.
- The object model for the master rule project includes static objects that are named `AgeRange`, `Gender`, `IncomeRange`, and `MaritalStatus`. These classes correlate to the commerce database contents.
- The values for any dynamic properties that using in the condition *<if condition, then action>* statements of your rules are based on the entire site, not on a particular merchant.
- Only classes displayed in the Blaze Advisor Builder with a red dot are eligible for use in the condition statements of your rules. The `internalUsePatternData` class is the one exception.
- Only one class is available for use in your action statements. This is the `productSelect` on `DB` class. It has a blue dot.
- There are four functions available for use. The four functions are:
 - a. `selectProduct`: Selects a product where the parent product number is null.
 - b. `filterProduct`: Filters a product where the parent product number is null.
 - c. `selectSKU`: Selects a product based on product reference number or stock keeping unit (SKU).
 - d. `filterSKU`: Filters a product based on product reference number or stock keeping unit (SKU).
- If you want to make use of timestamps, you should implement some checking mechanism to verify that the timestamp is not `NULL`. If it is `NULL`, and there is no checking, you will get a null pointer exception.

16.1.1.2 Worksheets

WebSphere Commerce Studio includes a number of simple worksheets to assist you in identifying the requirements for a personalized promotion campaign. You can find them in `\ibm\CommerceStudio\books\<locale>`

directory, where your language feature code should replace <locale> (for example en_US).

The following worksheets are provided with the Commerce Studio:

- Marketing campaign worksheet
This worksheet contains fields that correspond to the components that you need to identify for your campaign. The file name for this worksheet is mrktcamp.pdf.
- Audience worksheet
This worksheet contains columns for each audience you have targeted, and rows for each property currently available for the user class in the provided master rule project. The file name for this worksheet is called wkst.pdf.
- Product worksheet
This worksheet contains columns for each product being promoted, and rows for each property currently available for the product class in the provided master rule project. The file name for this worksheet is prodwkst.pdf.
- Default product recommendations worksheet
This worksheet is similar to the product worksheet with a different title. This identifies the product data used for default recommendations. The file name for this worksheet is deftwkst.pdf.

16.2 Personalization example

In this example, the ITSO sample store is running a promotion for the WebSphere CD collection to those registered shoppers who belong to IBM shopper group. If a user of the IBM shopper group purchases a WebSphere redbook, they will be prompted to purchase a WebSphere CD (cross-sell).

In this example, we will provide detailed instructions for each of the following high-level steps for implementing personalizations on the ITSO store:

1. Personalization worksheets.
2. Test the store before personalization changes.
3. Create the rule project and ruleflow.
4. Create the ruleset and assign task.
5. Compile the rule project.

6. Debug the rule project.
7. Modify the net.data macro associated with the selected VIEW task to display the recommendation.
8. Publish the project.
9. Configure personalization for WebSphere Commerce Suite.
10. Test the store after personalization changes.
11. Maintain the rule project.

16.2.1 Personalization worksheets

The following worksheets provide a summary of the target promotion. The personalization rule implementation is based on the information contained in the worksheets. The worksheets can be found in the c:\ibm\CommerceStudio\books\<locale> directory, on a system where the WebSphere Commerce Studio has been installed.

Table 34. Marketing campaign worksheet

Campaign item	Description
Objective	Drive the sales of WebSphere Redbook CDs through the use of a targeted merchandising strategy.
Target audience	Shoppers who belong to the IBM shopper group.
Products to be promoted	WebSphere Redbook CDs.
Web site location for personalization	Recommendations on checkout page.

Table 35. Product worksheet

Campaign item	Product description	Product description
Target Product	1	2
Product Name	Application Development CD collection	WebSphere CD collection
productID (generated by wcs)	345	657
parentProductID (generated by wcs)	10	12
SKU	21189-4	25486-9

Campaign item	Product description	Product description
shortDescription	Collection 1	Collection 1
onSpecial	N	N
inventoryLevel	350	217

16.2.2 Test the store before personalization changes

Prior to making the personalization changes to our store, we need to verify the behavior of the store. For example, if we are going to change the shopping interaction with users in the IBM shopper group, we should verify that this function is working properly as is. Also, we should verify that the products or categories of products that we would like to use in our personalization strategy are being displayed properly as is.

Complete the following steps:

1. If you have not already done so, create a shopper group in the WebSphere Commerce Administrator (ncadmin) named IBM.
2. Register a user named IBM. Add the IBM user to the IBM shopper group.
3. Register a user named Public.
4. Log on as user IBM, and purchase a WebSphere redbook. Take note of the shopping cart. After we add personalization later in this chapter, the shopping cart should be personalized to implement cross-sell.
5. Log on as user Public, and purchase a WebSphere redbook.

Now that we have established the behavior of the ITSO store, we can continue to the next section.

16.2.3 Create the rule project and ruleflow

This section provides instructions for creating the rule project, and ruleflow.

16.2.3.1 Create the rule project

Complete the following steps to create a rule project within the ITSO store in WebSphere Commerce Studio:

1. Launch the WebSphere Commerce Studio.
2. Open the project for the ITSO store.
3. Highlight the ITSO project folder.

4. From the menu bar, select **Tools -> Wizards -> Create Blaze Advisor Rule Project**.
5. When the Create Blaze Advisor Rule Project window appears, do the following:
 - a. Select **RecommendationMaster.adv**, as the master rule project, from the drop-down list.
 - b. Enter the name **itso** for the rule project to create, and then click **OK**.
6. The rules folder will appear within the itso project folder in WebSphere Studio project. Expand the **stores** folder. Select the **rules** folder, and then drag and drop the **rules** folder in the **ITSO** folder under stores.
7. Select the **rules** folder. Expand the **rules** folder, and then expand the **projects** folder to view the default files contained in the rule project, as seen in Figure 82.

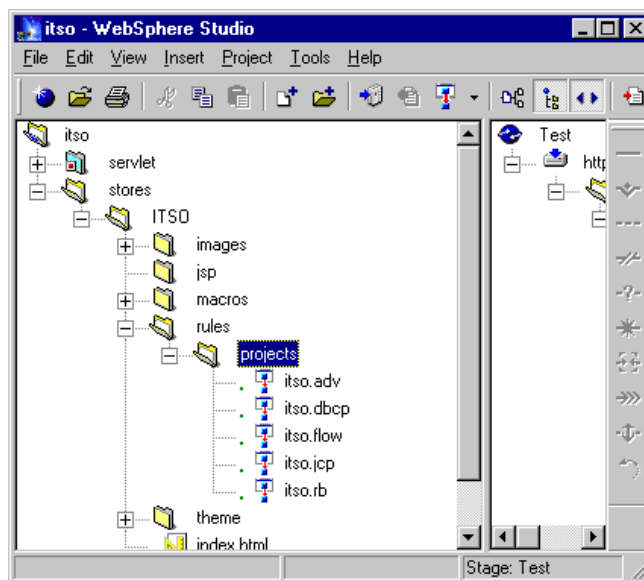


Figure 82. View rule project created in WebSphere Commerce Studio

16.2.3.2 Create the ruleflow

To create the ruleflow, complete the following steps:

1. Double-click the **itso.adv** file in the projects folder. This will start the Blaze Advisor Builder as seen in Figure 83.

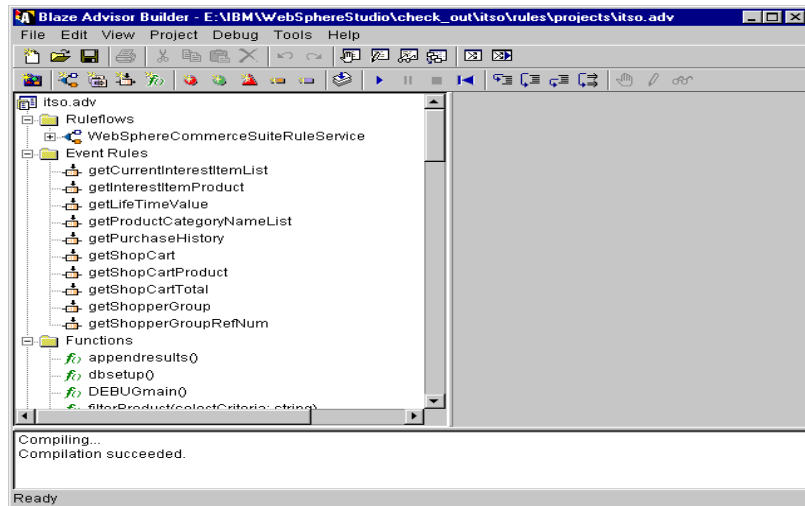


Figure 83. Blaze Advisor Builder

2. Select and expand **WebSphereCommerceSuiteRuleService** in the **Ruleflows** folder.
3. Double-click **Request_Service_Branch_With_Conditions**.
4. From the right-hand frame, click **New Branch**.
5. Enter the following in the Branch Condition 3 field:
ncRequest.service = "myrecommendation"
6. Double-click **WebSphereCommerceSuiteRuleService**.
7. A graphical view of the ruleflow is displayed in the RuleFlow Editor. From the RuleFlow Editor menu bar, click **Insert Task**.
8. Move the mouse pointer to **Branch 3**. A gray box indicates where the task will appear in the ruleflow. Click to add the ruleflow task.

16.2.4 Create the ruleset and assign task

Next we will create a rule to carry out the business logic inside the appropriate ruleset for the merchandising campaign. The following steps will guide us through converting information from the worksheets into rules, based on one of the templates provided. We will use the information from the target audience in the sample audience worksheet in Table 34 on page 383. We will implement a cross-sell rule, based on shopper group rule template.

To create a ruleset, complete the following steps:

1. Double-click the ruleflow **Task3** (created in the previous section) to open the Task Editor.
2. When the Task Editor window appears, do the following:
 - a. Enter MyRecommendationTask in the Task name field.
 - b. Under Implementation select **Ruleset**.
 - c. Under Choice select **New Ruleset**.
3. Enter myrecommendation in the Ruleset field. Select Return type **void**.
4. From the Ruleset Editor menu, click the **New from Template** icon as seen in Figure 84.

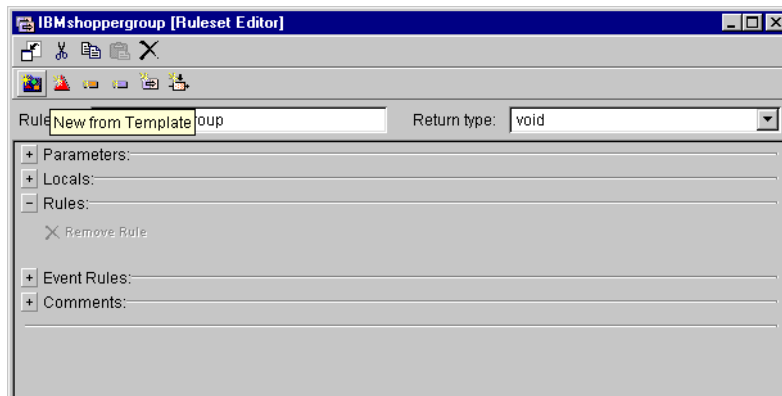


Figure 84. New from template

5. When the Template Chooser window appears, select the **Cross sell based upon the shopper's shopper group** rule template as seen in Figure 85, and then click **OK**. This will add the rule to the Ruleset Editor.

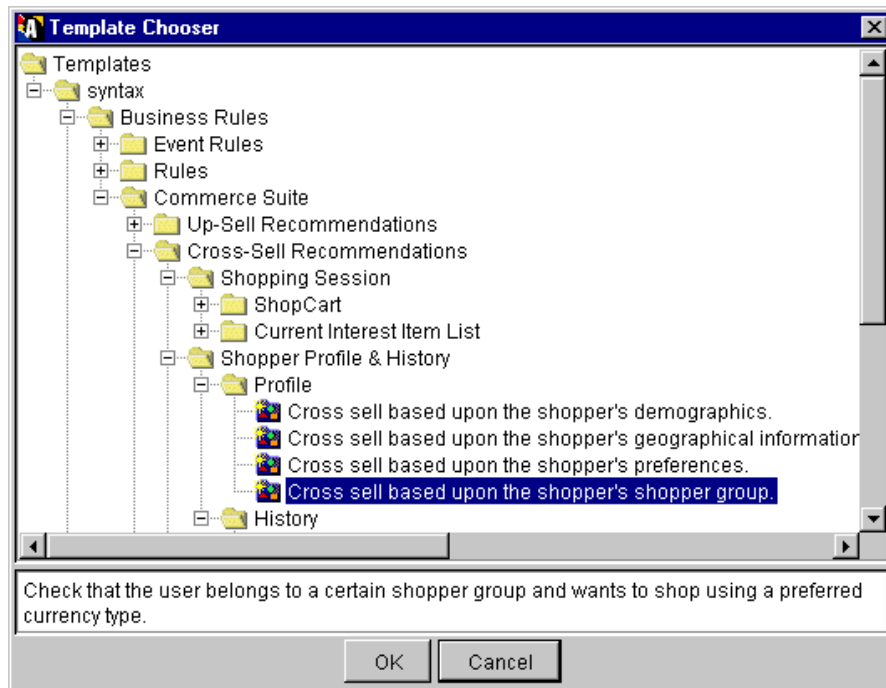


Figure 85. Cross sell based on shopper's shopper group template

6. Double-click the **Rule Name** text, which is highlighted in blue in the Rule field.
7. When the Replace Placeholder window appears, select the **rule1** text, and replace it with **IBMrules**, and then click **Replace**.
8. Set the Priority field to 0.
9. Edit the rule, in the rule definition area, so that the values match the product catalog and marketing campaign. For example:


```
if currentUser.shopperGroup = "IBM"
then {
    selectProduct("categoryName = 'WebSphere CDs'").
}
```
10. When complete, the rule should look like Figure 86.

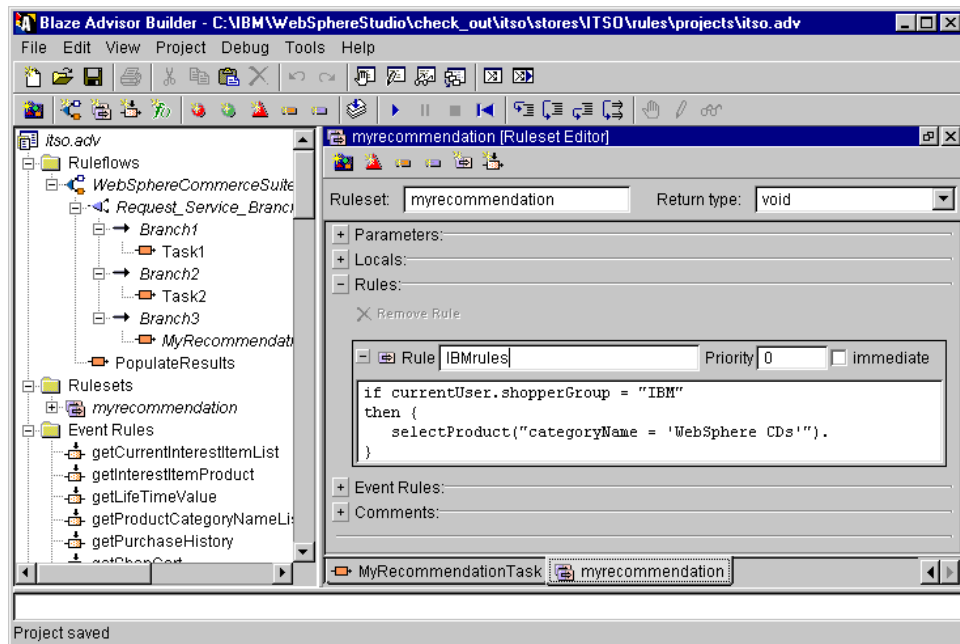


Figure 86. Ruleset editor - cross-sell rule

11. From the Blaze Advisor Builder menu bar, select **File -> Save Project**.

16.2.5 Compile the rule project

After creating the rules, we need to compile the rule project by completing the following steps:

1. From the Blaze Advisor Builder menu bar, select **Project -> Compile**.

This converts the rule project from the Blaze Advisor's Structured Rule Language to code for the server to interpret. Compiling also ensures that the syntax of the project is error-free. If you save the project with a syntax-error, you will see the message in Figure 87 the next time you start the Blaze Advisor Builder.

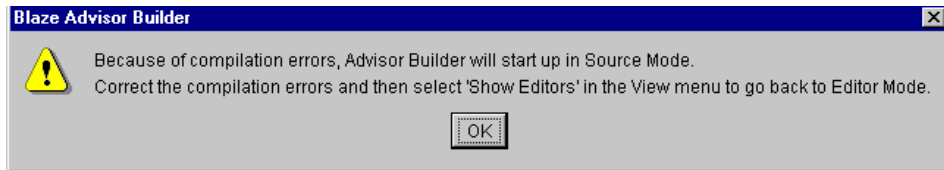


Figure 87. Syntax error found while opening rule project

2. If the compilation is successful, you should see the following message in the bottom panel:

Compiling...
Compilation succeeded.

We recommend that you compile the project frequently to make sure that no syntax errors exist before exiting from the project.

Note

A successful compilation does not guarantee that the project is error-free. Run the project after compiling it to test that it completes successfully. Potential problems include anomalies such as:

- Infinite loops
- Decision blocks that do not have branches to cover all possible outcomes.

Both problems listed could cause unexpected results with the WebSphere Commerce Suite.

3. From the Blaze Advisor Builder menu bar, select **File -> Save Project**.
4. Close the Blaze Advisor Builder.

16.2.6 Modify the net.data macro associated view task

In this section, we will modify the net.data macro associated with each view task that displays product recommendations. The macros include logic for processing and for displaying the recommended products. Table 36 identifies the default macros for the typical view tasks.

Table 36. Macros for view task

View task	Net.data macro
LOGON_DSP	login.d2w

View task	Net.data macro
CAT_DSP	This is a custom net.data macro that was generated during the store creation.
PROD_DSP	This is a custom net.data macro that was generated during store creation.
SHOPCART_DSP	shopcart.d2w
ORD_DSP_PEN	order.d2w

The following excerpts of code process information and then display the recommendations in a list. The net.data function DisplayRecommendationForShortList displays the following:

- Thumbnail graphic
- Product short description
- Link to its product display page

The macro handles a maximum of three recommendations. This sample code fragment can be used as is, by copying the code fragments listed in this document. In the ITSO shop, we modified the net.data macro shopcart.d2w to display the recommendation for the IBM shopper group.

1. Variables defined

The first segment of code defines the required variables. Insert the following segment into your net.data macro where you currently define variables:

```
StoreRefNum = merfnbr ? "$(merfnbr)" : "-1"
SP1 = ProdRec1 ? "$(ProdRec1)" : "-1"
SP2 = ProdRec2 ? "$(ProdRec2)" : "-1"
SP3 = ProdRec3 ? "$(ProdRec3)" : "-1"
```

2. Function

The next code segment defines the function that performs the logic to process the product reference numbers returned from the Blaze Advisor server. Insert the following segment into your net.data macro where you have defined your other functions:

```
%FUNCTION(DTW_ODBC) DisplayRecommendationForShortList(IN theProduct, IN
theMerchant) {
    SELECT prnbr, prsdsc, prthmb, prrfnbr, pmenbr
    FROM product
    WHERE prrfnbr=$(theProduct)
    AND pmenbr=$(theMerchant)
```

```

%REPORT {
    <TR>
        <TD><IMG SRC="$(V_prthumb)"></TD>
        <TD><A HREF="/cgi-bin/ncommerce3/ProductDisplay?pmenbr=
$(V_pmenbr)&prrfnbr=$(V_prrfnbr)">$(V_prnbr)</A></TD>
        <TD>$(V_prsdesc)</TD>
    </TR>
%}
%MESSAGE {
    100 : { %} : continue
    default: { %} : continue
%}
%}

```

3. Function call

The last segment of code calls the function and displays the results. Insert the following segment into the HTML report section where you want the recommendations to appear:

```

%HTML(REPORT) {
.....

%if(SP1 != "-1")
    <P><B>Would you be interested in... </B></P>
    @DisplayRecommendationForShortList(SP1, StoreRefNum)
    @DisplayRecommendationForShortList(SP2, StoreRefNum)
    @DisplayRecommendationForShortList(SP3, StoreRefNum)
%endif

```

16.2.7 Publish the project

This section provides instructions for publishing the project with specific information for personalization. To publish the base store to the target host, complete the following steps:

1. Verify that your WebSphere Application Server is started.
2. From the left-hand frame select <your_project> -> right-click -> **Publish whole Project**.
3. When the Publishing Options window appears, click **OK**.
4. When the Files to publish window appears, click **OK**.

5. When the Commerce Suite server logon window appears, enter the following:
 - User name: ncdadmin
 - Password: <your_password>
 - Click **OK**

Note

The first time you publish you will get a message stating that folders need to be created. This will also happen if you create new folders to publish.

6. Change access permissions

On the target host Commerce server, change the access permissions of the files published by typing the following:

```
# su - db2inst1
$ chmod -R 755 /projects
```

Note

You need to change the access permissions every time you publish your project to the AIX WebSphere Commerce server.

16.2.8 WebSphere Commerce Suite personalization configuration

This section provides instructions for configuring personalization within the WebSphere Commerce Suite.

16.2.8.1 WebSphere Commerce Configuration Manager

To enable the Commerce Suite server for personalization, configure the Blaze Advisor Rule Server, and Rule Service by doing the following:

1. Start the WebSphere Commerce Configuration Manager.
2. Select the instance for which you are enabling personalization, and then click **Settings**.
3. Click the **Rule Server** tab.

Configure the Rule Server by doing the following:

- a. Ensure that the **Enable Rule Server** check box is selected.
- b. In the Maximum Recommendations field, enter a limit for the maximum number of product recommendations that can be requested in a single

- invocation of the recommendation service; this prevents large requests from slowing down the Advisor rule server performance. The default value is 100, and the maximum value is 65536. The Commerce Suite server uses the default if you do not enter a value, and the maximum if you enter a value higher than the maximum (for example 1000).
- c. In the User Java Classpath field, enter any additional user-specified classpath values. These are necessary when importing Java classes and .jar files. In our example, we left this field empty.
4. Click the **Rule Services** tab.
Configure the Rule Services by doing the following:
 - a. In the Rule Service field, enter `itso`. This name must match the name of the rule service name exactly, including case.
 - b. In the Advisor Project field, enter the name of the advisor project that implements the rule service. For example, `itso.adv`. This is the directory where the Advisor project has been published.
 - c. In the Interface Object Classname field, enter the classname of the interface object for this rule service. Enter `RecommendProductsData` for the default recommend products service.
 - d. In the Number of Constructor Arguments field, enter the number of constructor arguments that are appropriate for the interface object. Enter 4 for the default Recommend Products service.
 - e. Click **Add** to add the rule service to the Commerce Suite configuration file.
 5. Click **OK** to save.

Note

The Configuration Manager writes the settings to the following files:

```
/usr/lpp/CommerceSuite/instance/<instance_name>/config/wcs_advisor.server  
/usr/lpp/CommerceSuite/instance/<instance_name>/config/ncommerce.ini
```

6. After the Configuration Manager settings have been changed, click **OK**.

16.2.8.2 Modify the `wcs_advisor.server` file

In order to publish the rule project to the directory specified in 13.2.2, “Configuring the target publishing host - AIX platform” on page 277 we need to modify the `wcs_advisor.server` file manually. When any settings are saved from the Configuration Manager, the `BaseFilePath` is overwritten with the default path.

To modify the `wcs_advisor.server` settings, complete the following steps:

1. On the WebSphere Commerce server (AIX platform), log in to AIX as user root, and start a terminal session.

2. Change to the directory of the `wcs_advisor.server` file found in the WebSphere Commerce instance directory:

```
# cd /usr/lpp/CommerceSuite/instance/<instance_name>/config
```

3. Backup the original `wcs_advisor.server` file:

```
# cp wcs_advisor.server wcs_advisor.server.orig
```

4. Modify the `wcs_advisor.server` file:

a. Find the tag **<BaseFilePath>** and change the path:

From: `/usr/lpp/CommerceSuite/rules/projects/`

To: `/projects/stores/ITSO/rules/projects/`

Note

The new `BaseFilePath` is the same as the `storesDocRoot` defined in 13.2.2, “Configuring the target publishing host - AIX platform” on page 277.

The `BaseFilePath` will get overwritten with the default settings every time you save settings through the Configuration Manager.

b. Find the tag **<Project>**. Change the project value to the name of the studio project. For example, `itso.adv`.

16.2.8.3 WebSphere Commerce Administrator

The product recommendation rule service is administered from the WebSphere Commerce Administrator (`ncadmin`). In this example, we will personalize the shopcart display page for the ITSO store. This overridable function is defined at the store level.

Note

This step requires that you have published a store previously. The store that you publish does not have to be complete. The Commerce Suite Administrator will not let you select a store or rule services unless a store exists on the Commerce Suite server.

Define VIEW task for displaying recommendations

To define the `VIEW` task for displaying the recommendations, complete the following steps:

1. Launch the WebSphere Commerce Administrator (ncadmin).
2. From the task bar, click **Site Manager**.
3. Click **Task Management**.
4. When the Task Management window appears, do the following:
 - a. Select **VIEW** from the Select Task Type pull-down list.
 - b. From the bottom frame, scroll through the list of tasks and select **SHOPCART_DSP**, and then click **SAVE**.
5. Click **Task Assignment**, and select **Overridable Function**.
6. When the Overridable Function Assignment window appears, do the following:
 - a. Select **ITSO** from the Stores list.
 - b. Select **TaskDisplay** from the overridable functions list.
 - c. Click **Update** to save.

Note

This task will attach the ND_RecommendProducts overridable function to the selected view task. The ND_RecommendProducts overridable function serves as a generic interface for the Commerce Suite personalization framework and invokes the Recommend Products rule service.

7. Click **Task Assignment**.
8. Click **Macro** and assign the macro. In this example, we selected **/ITSO/macro/shopcart.d2w** that we have modified. Click **Save**.

Administer product the recommendation rule service

To administer the product recommendation rule service, complete the following steps:

1. Open the Commerce Suite Administrator.
2. From the task bar, click **Site Manager** -> **Personalization**.
3. From the Rule Service drop-down list, select either **Recommend Products** which is the default rule service created for the Commerce Suite personalization framework, or any other custom rule services appearing in the list (for example itso).
4. From the Store Name drop-down list, select either **MALL** to implement personalization for all stores in the mall, or <store_name> to implement. We

selected ITSO store as we need to implement the personalization at store level only.

5. Select and configure the view task by doing the following:
 - a. From the View Tasks drop-down list, select the **SHOPCART_DSP** view task.
 - b. In the Ruleset Name field, enter myrecommendation for the ruleset that is used with the selected view task.

Note

This is not really the ruleset name but the value we entered in the branch condition (ncRequest.service = "myrecommendation").

- c. In the Number of Results field, enter the maximum number of results that you want to receive from the rule server when this ruleset is called (for example 3).
- d. Select the **Enabled** check box.
- e. Click **Update**.

16.2.8.4 Restart the IBM HTTP Server

To restart the IBM HTTP Server, do the following:

1. Change the server directory:

```
# cd /usr/HTTPServer/bin
```

2. Restart the server:

```
# ./apachectl restart
```

16.2.8.5 Restart the WebSphere Commerce Server

For the changes to take effect, you must restart the WebSphere Commerce Suite server. To restart the Commerce Suite server, do the following:

1. Launch the WebSphere Commerce Configuration Manager.
2. When prompted, enter your Configuration Manager user ID and password.
3. From the list of instances, select the instance you want to stop and click **Stop**. The state should indicate Inactive.
4. With the same instance selected, click **Start**. The state should indicate Active.

16.2.9 Test the store after the personalization changes

Test your store to ensure that product recommendations are working correctly. To complete the testing, make a number of visits to your store, using a number of different users.

When testing, use shoppers that have profiles that meet the different criteria that you defined for various merchandising campaigns. We should have a shopper who belongs to IBM shopper group. Make sure that each shopper views the different pages for which you have enabled personalization. For instance, if you have provided recommendation strategies on both the category display page and the checkout page, make sure that both of these pages are displayed to see what happens when the different shoppers view each page. If the shoppers have different profiles, and your rule project is robust, then they should see different product recommendations. If they see the same products consistently, you may want to determine if there are more rules that can be added, or conditions that can be changed to produce results that are more appropriate for each shopper.

16.2.10 Maintain the rule project

This personalization implementation allows you to dynamically edit your rule project. Rule maintenance requires the use of the development environment, but you do not have to shut down the entire site every time that you want to change your rules. You have the option of resetting the rule service instead of stopping the Commerce Suite server. Resetting the rule service forces the Blaze Advisor to use the updated rule project the next time that it calls the server. Rule processing current at the time that the rule service is updated will continue to return results that are based upon the previous iteration of the rule project. In this way, your customers see a seamless shopping trip, and you do not risk losing business due to server down time. To update a rule service without shutting down the site, do the following:

1. Launch the WebSphere Commerce Administrator.
2. From the task bar click **Site Manager**, and then **Rule Services**.
3. From the Rule Service drop-down list, select rule service that corresponds to the rule project that you have changed.
4. Click **Reset**.

16.3 Personalization debug tips

This section provides debug tips that we discovered while testing this feature of the product.

1. Compile errors

Any errors in the code encountered during the compile step are displayed in the message window across the bottom of the Advisor Builder interface. Generally, errors are caused due to mistakes in typing the name of some item or property, but they may also be caused by attempting to perform unsupported actions on some variable. For instance, to attempt mathematical operations on a string would result in an error.

Note

Double-click the error message to open the rule project to the location of the error. If the cause of the error is difficult to determine, you can use the debugging tools that are incorporated in the Advisor Builder. For example, you can use the tools to insert break-points into your project to determine where the problem is. Refer to the Blaze Advisor Builder documentation for more information on debugging.

2. No recommendation list displayed

When you test your rule services on the store, you will find that no recommendation list appears on the designated page. In this case, you should make sure of these things:

- Has the rule server properly started?
- Do the personalization rules return any recommendation?

We can easily check out these two possible problems by looking at the log file of WebSphere Commerce Suite. To enable the log function, we need to change of the parameter MS_LOGLEVEL from 0 to 3 in the ncommerce.ini/ncommerce.conf file and restart the WebSphere Commerce server.

3. Configuration Manager change

The default advisorDocRoot is /usr/lpp/CommerceSuite/rules/projects/ in the WebSphere Commerce Studio. Also the default <BaseFilePath> tag in the wcs_advisor.server file is /usr/lpp/CommerceSuite/rules/projects/.

When settings are modified and saved using the WebSphere Commerce Configuration Manager, any manual changes to the wcs_advisor.server file for the <BaseFilePath> are reset to the default.

Chapter 17. Back-end integration using MQSeries XML messages

When implementing an enterprise-out Web site, seamless integration with back-end systems is very important. The MQSeries adapter is a component of the WebSphere Commerce Suite that enables integration with back-end systems using MQSeries messages.

This chapter is organized into the following sections:

- WebSphere Commerce Suite integration with MQSeries
- MQSeries client/server runtime environment
- MQSeries adapter enablement
- MQSeries XML message example
- MQSeries adapter debugging tips
- Where to find more information

17.1 WebSphere Commerce Suite integration with MQSeries

The WebSphere Commerce Suite MQSeries Adapter provides an interface to the MQSeries server to send and receive message from the MQSeries server queue. The messages provided with the MQSeries adapters are available as standard MQSeries messages, and a subset of the messages are in MQSeries XML format. These messages are provided to handle data inbound and outbound for the WebSphere Commerce server. The messages invoke a specific behavior within the WebSphere Commerce Suite server, such as customer registration, and change product price.

The format of the XML messages consists of a set of XML elements defined with specific DTD files for each XML message. Each DTD contains one or more command files, identified by a .mod file extension. Table 37 provides a description of the supported MQSeries XML messages.

Table 37. Six supported XML messages offered with the MQSeries adapter

Message Name	Message Type	DTD File	Common File
Create_NC_Customer	Inbound	Create_NC_Customer_10.dtd	NCCCommon.mod NCCustomer_10.mod (source file) NCCustomer_10.mod (detailed file)

Message Name	Message Type	DTD File	Common File
Report_NC_PurchaseOrder	Outbound	Report_NC_PO_10.dtd	N/A
Update_NC_Customer	Inbound	Update_NC_Customer_10.dtd NCCCommon	NCCCommon.mod NCCustomer_10.mod (source file) NCCustomer_10.mod (detailed file)
Update_NC_OrderStatus	Inbound	Update_NC_OrderStatus_10.dtd	NCCCommon.mod
Update_NC_ProductInventory	Inbound	Update_NC_ProductInventory_10.dtd	NCCCommon.mod
Update_NC_ProductPrice	Inbound	Update_NC_ProductPrice_10.dtd	NCCCommon.mod

17.1.1 MQSeries inbound XML

An inbound message is a message that the WebSphere Commerce Suite receives from an external application. The XML messages are encoded in UTF-8 and UTF-16 format.

The MQSeries adapter supports the following inbound messages:

1. XML format
 - Create_NC_Customer
 - Update_NC_Customer
 - Update_NC_OrderStatus
 - Update_NC_ProductPrice
 - Update_NC_ProductInventory
2. Commerce Suite format
 - Customer New
 - Customer Update
 - Order Status Update
 - Product Price Update
 - Product Quantity Update

17.1.2 MQSeries outbound XML

An outbound message is a WebSphere Commerce Suite generated message that can be sent to an external system. The MQSeries adapter supports one outbound message in XML format, called Report_NC_PurchaseOrder. This

XML message is encoded in UTF-8 format. The adapter also supports one outbound message in the Commerce Suite format, called Order Create. The outbound messages contain order information copied from the WebSphere Commerce server to external systems, where order fulfillment processes take place.

17.2 MQSeries client/server runtime environment

This section provides an overview on enabling a connection between legacy systems, and a WebSphere Commerce server using MQSeries V5.1 for AIX as middleware for message communication between the two systems.

MQSeries can be configured in either of the following methods:

1. MQSeries client/server-to-server

For the client/server-to-server interface, the MQSeries server is installed on the same machine as the WebSphere Commerce server.

2. MQSeries client-to-server

For the client-to-server interface, an MQSeries client is installed on the same machine as your WebSphere Commerce server. The MQSeries client is configured to communicate with the MQSeries server.

In this example, we will implement an MQSeries client-to-server environment on the AIX platform.

Note

Although the installation and configurations instructions are for AIX, many of the steps are the same between AIX and Windows NT.

17.2.1 MQSeries server V5.1 for AIX

This section provides instructions for installing, configuring, and verifying the MQSeries server.

For a more detailed description of the instructions, refer to the following platform specific-guides. The following documents can be found at: <http://www.ibm.com/software/ts/mqseries/library/manualsa/>.

- *IBM MQSeries for AIX, Quick Beginnings V5.1*, GC33-1867-02
- *MQSeries Clients*, GC33-1632-07
- *IBM MQSeries for NT, Quick Beginnings V5.1*, GC34-5389

17.2.1.1 MQSeries installation prerequisites

Prior to the installation of the MQSeries server, complete the following steps:

1. Log in to AIX as root, and start terminal session.
2. Create the MQSeries file systems, as described in Table 38, by completing the following high level steps:
 - a. Create a logical volume.
 - b. Create a file system, and mount point.
 - c. Mount file system.
 - d. Change the file system size.

Table 38. MQSeries file systems

Logical volume	File system mount point	File system size
mqmlv	/var/mqm	100 MB
mqmloglv	/var/mqm/log	50 MB
mqmerrlv	/var/mqm/errors	2 MB

3. Verify that the file system space of /usr has at least 30 MB of free space. If it does not, increase the size of /usr accordingly.

```
# df /usr
```

4. Create the group mqm by typing the following:

```
# mkgroup mqm
```

5. Create the user mqm by typing the following:

```
# mkuser pgrp=mqm mqm
```

6. Set the default password for user - mqm

When creating a user, the password is set to *, for invalid. It is necessary to set the password by typing the following:

```
# passwd mqm
```

```
Changing password for "mqm"
```

```
mqm's New password: mqm
```

```
Enter the new password again: mqm
```

7. Change password during initial logon - mqm.

```
# login mqm
```

```
mqm's Password: mqm
```

```
mqm's New password: mqm
```

Enter the new password again: mqm

8. Add user root to the mqm group by using smitty, by doing the following:
 - a. # smitty group
 - b. Select **Change / Show Characteristics of a Group**
 - Group NAME: mqm
 - USER List: mqm, root
 - Press Enter.
9. Press F10 to return to the system prompt.

17.2.1.2 MQSeries server installation

To install the MQSeries server, complete the following steps:

1. Log in to AIX as root, and start a terminal session.
2. Insert the MQSeries CD.
3. Mount the CD-ROM, by typing the following:

```
# mount -r -v cdrfs /dev/cd0 /mnt
```
4. Start the install:

```
# cd /mnt/install.images/MQSeriesv51
# smitty install_latest
```
5. When the Install and Update from LATEST Available Software window appears, enter the following:
 - INPUT device: ./
 - Press Enter.
6. When the SOFTWARE to install window appears: press F4 for a list, and select the packages listed in Table 39. Once all the packages have been selected using F7, press Enter.

Table 39. MQSeries server installation packages

Package name	Description
mqm.Client.Bnd	MQSeries client bundles
mqm.Server.Bnd	MQSeries server bundles
mqm.base	MQSeries base kit for client and server
mqm.client	MQSeries client for AIX
mqm.java	MQSeries Java base, bindings, runtime

Package name	Description
mqm.man	MQSeries man pages
mqm.msg.en_US	MQSeries messages for U.S. English
mqm.Server	MQSeries server for AIX

7. Press Enter to continue.
8. When the Are You Sure? message appears, press Enter.
9. View the installation log to verify that all the packages were installed successfully.
10. Press F10 to return to the system prompt.

17.2.1.3 MQSeries server configuration

Once the MQSeries server is installed, the server must be configured by completing the following steps:

1. Log in as user mqm by typing the following:

```
# su - mqm
```

2. Change directories by typing the following:

```
$ cd /usr/mqm/bin
```

Windows NT

The directory is: c:\ibm\mqseries\bin.

3. Create the default queue manager; for example, type the following:

```
$ crtmqm -q QM1.WCS
```

```
MQSeries queue manager created.
```

```
Creating or replacing default objects for QM1.WCS.
```

```
Default objects statistics : 29 created. 0 replaced. 0 failed.
```

```
Completing setup.
```

```
Setup complete.
```

4. Start the default queue manager by typing the following:

```
$ strmqm
```

```
MQSeries queue manager 'QM1.WCS' started.
```

5. Enable the MQSC commands from the command line by typing the following:

```
$ runmqsc
```


Starting MQSeries Commands. (will not return to prompt)

6. Create the queues for the queue names listed in Table 40 by typing:

```
define qlocal (<queue_name>)
```

Where <queue_name> is replace with INBOUND, OUTBOUND, and ERROR.
Repeat this command for each queue.

Table 40. WebSphere Commerce MQSeries - queues

Queue name	Description
INBOUND	WebSphere Commerce inbound queue
OUTBOUND	WebSphere Commerce inbound queue
ERROR	WebSphere Commerce error queue

7. Create a server-connection channel by typing the following command:

```
DEFINE CHANNEL(CHANNEL1) CHLTYPE(SVRCONN) TRPTYPE(TCP) MCAUSER('')
```

The following message should be displayed after typing the previous command:

```
MQSeries channel created.
```

8. To stop MQSC, press Ctrl-D or type end. The following message should be displayed:

```
All valid MQSC commands were processed.
```

9. Configure the system to start channels.

Configure the inetd daemon on the server, so that inetd will start the MQI channels.

- a. Log on as root.

- b. Add the following line to the /etc/services file:

```
MQSeries 1414/tcp
```

- c. Add the following line (case sensitive) to /etc/inetd.conf file to call the program amqcrsta:

```
MQSeries stream tcp nowait mqm /usr/mqm/bin/amqcrsta amqcrsta -m QM1.WCS
```

- d. The updates are active after you issue the following command from the root user ID:

```
# refresh -s inetd
```

17.2.1.4 MQSeries server verification

Once the server is installed and configured, we need to verify that the configuration, queue manager, and queue is working properly. We will verify the configuration by using a sample application to put a message onto the queue, and to read the message from the queue:

1. Log in as user mqm, and change to the directory of the samples.

```
# su - mqm
$ cd /usr/mqm/samp/bin
```

2. Put a test message on the queue by typing the following:

```
$ ./amqsput INBOUND
This is a test message on the INBOUND queue
```

Note

Press enter twice after entering the test message.

3. Get message from the queue by typing:

```
$ ./amqsget INBOUND
Your test message should be displayed.
```

4. The server verification is complete.

17.2.2 MQSeries client V5.1 for AIX

The procedure to install the MQSeries client is very similar to the MQSeries server installation.

For a more detailed description of the instructions, refer to the following platform-specific guides. The following documents can be found at: <http://www.ibm.com/software/ts/mqseries/library/manualsa/>.

- *IBM MQSeries for AIX, Quick Beginnings V5.1*, GC33-1867-02
- *MQSeries Clients*, GC33-1632-07
- *IBM MQSeries for NT, Quick Beginnings V5.1*, GC34-5389

To install, configure, and verify the the MQSeries client, complete the following sections.

17.2.2.1 MQSeries installation prerequisites

The MQSeries client prerequisites are the same as for the server. Refer to 17.2.1.1, “MQSeries installation prerequisites” on page 404 for detailed instructions.

17.2.2.2 MQSeries client installation

The MQSeries client installation procedure is the same as the server. Refer to 17.2.1.2, “MQSeries server installation” on page 405 for detailed instructions. During the MQSeries client installation, select the client packages listed in Table 41.

Table 41. MQSeries client installation packages

Package name	Description
mqm.Client.Bnd	MQSeries client bundles
mqm.base	MQSeries base kit for client and server
mqm.client	MQSeries client for AIX
mqm.man	MQSeries man pages.
mqm.msg.en_US	MQSeries messages in U.S. English

17.2.2.3 MQSeries client-to-server configuration

Once the MQSeries client is installed, it needs to be configured to communicate with the MQSeries server.

1. On the MQSeries client system, log in as user mqm by typing the following:

```
# su - mqm
```

2. Modify the .profile with a text editor, and add the following line:

```
export MQSERVER=CHANNEL1/TCP/'<server-address>(<port>)'
```

Where <server-address> is the TCP/IP hostname of the server, and (<port>) is optional and is the TCP/IP port number the server is listening on, for example 1414.

For example:

```
export MQSERVER=CHANNEL1/TCP/'mqsrv1(1414)'
```

3. Return to user ID root, by typing:

```
$ exit
```

17.2.2.4 MQSeries client-to-server verification

Before proceeding to the next section, verify the communication between the MQSeries client and server.

1. On the MQSeries client system, log in as user mqm, and change to the directory of the samples:

```
# su - mqm
$ cd /usr/mqm/samp/bin
```

2. Put a test message on the queue by typing the following:

```
$ ./amqsputc INBOUND
```

This is a test message from the client on the INBOUND queue

Note

Press enter twice after entering the test message.

3. On the MQSeries server system, get a message from the queue by typing:

```
# su - mqm
$ cd /usr/mqm/samp/bin
$ ./amqsget INBOUND
```

Your test message should be displayed.

4. The client-to-server verification is complete.

17.3 MQSeries adapter

This section provides instructions for configuring the MQSeries adapter for use with the WebSphere Commerce server.

17.3.1 Pre-configuration of MQSeries adapter

Prior to configuring the MQSeries adapter, you must meet the following pre-configuration requirements:

- MQSeries client/server or server needs to be started before the WebSphere Commerce Suite instance is started.
- If you are using MQSeries client configuration, all the required channels must be defined and you must identify the channel name MQSeries.
- Read and write authorities need be to be granted to the database user logon to the queue manager and queues specified in the ncei_ece

configuration file. The ncei_ece file is created by the ncmqbridge in subsequent steps. For example, add the DB2 instance owner to the mqm group or grant the user mqm read, and write authority to the database.

17.3.2 Enable MQSeries adapter

To enable the MQSeries adapter, complete the following steps:

1. Stop the WebSphere Commerce Suite instance before executing the ncmqbridge utility.
2. Change to the directory containing the ncmqbridge executable:

```
# cd /usr/lpp/CommerceSuite/server/admin
```

Windows NT

The directory on Windows NT is: c:\ibm\wcs\server\admin

3. Execute the ncmqbridge utility. For example, type the following command:

```
# ./ncmqbridge wcsaix 18000 mqsrv.itso.ra1.ibm.com INBOUND OUTBOUND ERROR  
QM1.WCS db2inst1 client CHANNEL1
```

Syntax

```
ncmqbridge <instance_name> <port_number> <mq_server_name>  
<inbound_queue_name> <outbound_queue_name> <error_queue_name>  
<queue_manager_name> <mqbridge_userid> <client/server>  
<channel_name> <server_port_number>
```

For details on the parameters of the ncmqbridge, refer to the online help for the WebSphere Commerce Suite or the *WebSphere Commerce Suite V4.1 MQSeries Adapter Guide*, found on the product CD.

Note

The NC_MQ_BRIDGE_USERID must have WCS DB2 access permissions.

4. Start the WebSphere Commerce Suite instance.

17.3.3 Access control for the MQSeries adapter

Once the MQSeries adapter is enabled, the following commands are automatically available to the WebSphere Commerce Suite administrators:

- AdminRegisterNew

- AdminRegisterUpdate
- OrderItemStatusCmd
- OrderStatusCmd
- ParseMsg
- ProdPriceUpdateCmd
- ProductUpdateQuantity
- SendOrderMsg
- SendMQUserMsg
- SendXMLOrderMsg

These commands are available to be executed from the WebSphere Commerce server that processes the MQSeries adapter messages. The WebSphere Commerce server requires that the user ID in the NC_MQ_BRIDGE_USERID parameter has WebSphere Commerce administrator privileges. Ensure that the user ID in the NC_MQ_BRIDGE_USERID parameter is specified in the Site Administrators group within the Access Group Assignment.

17.3.3.1 Access group assignment

To set the access group assignment, complete the following steps:

1. Start the WebSphere Commerce Administrator (ncadmin).
2. Click **Site Manager -> Access Control**.
3. Enter the administrator user information, and then click **Save**.
4. Click **Access Assignment**.
5. Select **Site Administrator**, click **Add Group**, and then click **Save**.

17.3.4 Enable MQSeries XML messages

This section describes the procedure to enable inbound and outbound MQSeries messages.

17.3.4.1 Enable outbound messages

To enable outbound MQSeries messages in XML format, we need to update the task assignment for the process task, and overridable function within the WebSphere Commerce Administrator (ncadmin).

Configure process task: EXT_ORD_PROC

These steps are necessary to update the task assignment:

1. Start the WebSphere Commerce Administrator (ncadmin).
2. Select **Site Manager**, and then click **Task Management**.

3. When the Task Management form appears, select **Task Type** from the drop-down list, and select **PROCESS**. The Task Name list appears in the bottom frame.
4. From the Task Name list in the bottom frame, click **EXT_ORD_PROC** to fill-in the task.
5. If you wish to change the scope of the task, select **Mall** or **Store** from the Scope drop-down list, and then click **Save**.
6. On the task bar, click **Task Assignment**.
7. When the Overridable Function Assignment form appears, select `<your_store_name>`.
8. Scroll down to the **Overridable Functions** list, then select **ECEExtXMLOrdProc**, and then click **Update**.

The assignment record has been successfully added to the database.

Note

To enable the XML-supported outbound message, Report_NC_PurchaseOrder, select **ECEExtXMLOrdProc**.

To enable the Commerce Suite format outbound message, Order Create, select **ECEExtOrdProc**.

9. Stop and restart the WebSphere Commerce Suite server.

17.4 MQSeries XML message example

In this section, we will use the ITSO store to implement inventory level checking by using MQSeries XML messages with the WebSphere Commerce Suite.

17.4.1 WebSphere Commerce inventory level checking example

In the example, the information of inventory level is generated to simulate a back-end system. In a production environment, the integration with back-end systems will look similar to Figure 88.

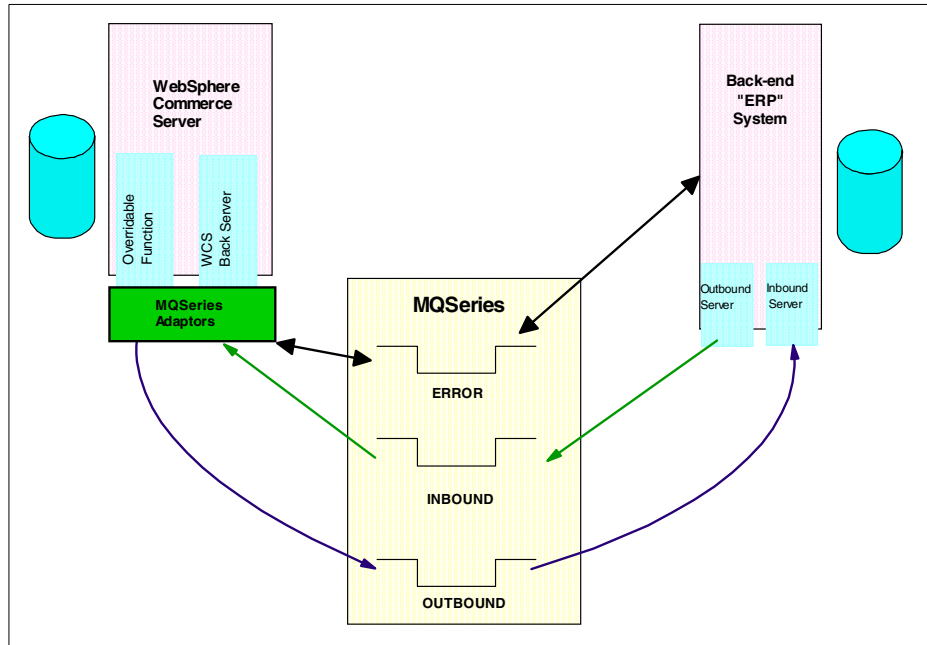


Figure 88. WebSphere Commerce MQSeries Integration to back-end system

17.4.1.1 Example message flow explanation

This section provides an explanation of how MQSeries XML messages interact with WebSphere Commerce for the ITSO store.

1. The shopper creates an order within the ITSO store.
2. An XML message is sent from the WebSphere Commerce server, based on the order details, to the MQSeries adapter.
3. The MQSeries adapter is configured to send the MQSeries message in XML format to the MQSeries server default queue manager QM1.WCS, OUTBOUND queue.
4. The back-end system will get the XML message from the OUTBOUND queue.
5. The back-end system will perform the following actions:
 - The back-end system will deduct the inventory level for the ordered product, based on the received XML message generated from the WebSphere Commerce server.
 - The back-end system creates an XML message that reflects inventory changes to be put on the INBOUND queue. The back-end system must

follow the XML message DTD format of Update_NC_ProductInventory. Refer to the *WebSphere Commerce Suite MQSeries Adapter Guide*, found on the product CD, for specific details about the Update_NC_ProductInventory XML message DTD format.

6. The WebSphere Commerce Suite will get the XML message from INBOUND queue and process the update within the WebSphere Commerce database.
7. The WebSphere Commerce server accepts the Update_NC_ProductInventory XML message and updates the inventory level for the particular product within the PRODUCT database table of the WebSphere Commerce database server.
8. The shopper can view the inventory level of products at the product page of the ITSO store.

17.4.1.2 Outbound message verification

In the previous section, we enabled XML messages for the WebSphere Commerce server. Now we can test the OUTBOUND message queue by viewing messages generated from the ITSO store by creating an order. If it is successful, we will be able to view the message added to the OUTBOUND queue.

Browse message

To browse the OUTBOUND message queue, complete the following steps:

1. Create an order for the ITSO store.
2. On the MQSeries server system, log in as user mqm by typing the following:

```
# su - mqm
```

3. Browse the OUTBOUND message queue, by typing the following:

```
$ cd /usr/mqm/samp/bin
```

```
$ ./amqsgbr OUTBOUND
```

Sample outbound XML message

Here is an example of a section of an outbound XML message that follows the Report_NC_PurchaseOrder format. The message below is a partial sample of the type of information in the XML formatted message:

```
<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE Report_NC_PurchaseOrder SYSTEM  
"Report_NC_PO_10.dtd"><Report_NC_PurchaseOrder version="1.0"><ControlArea><Verb  
value="Report"> </Verb><Noun value="NC_PurchaseOrder">  
</Noun></ControlArea><DataArea><ReportPO><ReportPOHeader><OrderNumberByMerchant>13  
816</OrderNumberByMerchant><OrderNumberByNC>13816</OrderNumberByNC><DateTim
```

```
nce><PlacedDate>20000609</PlacedDate><PlacedTime>183058</PlacedTime></DateTimeReference><TotalPriceInCurrency="USD"><TotalNetPrice>30.00</TotalNet
```

As you can see, the outbound message contains the order details for the transaction, including information about inventory.

17.4.1.3 Inbound message verification

The XML message, Update_NC_ProductInventory, contains information about product SKU number and inventory level. The Update_NC_ProductInventory message uses the XML format and follows Update_NC_ProductInventory_10.dtd. This file DTD format would need to be implemented by the message generation from a back-end system. This DTD files are located in the following directory:

- AIX: /usr/lpp/CommerceSuite/xml/MQSeriesAdapter
- Windows NT: c:\ibm\CommerceSuite\xml\MQSeriesAdapter

Sample inbound XML message

Here is an example of the Update_NC_ProductInventory XML message:

```
<?xml version="1.0"?>
<!DOCTYPE Update_NC_ProductInventory SYSTEM
"e:\mq\Update_NC_ProductInventory_10.dtd">
<Update_NC_ProductInventory version="1.0">
  <ControlArea>
    <Verb value="Update">
      "Update"
    </Verb>
    <Noun value="NC_ProductInventory">
      "By Cinderella"
    </Noun>
  </ControlArea>
  <DataArea>
    <ProductInventoryInfo>
      <ProductNumberByMerchant>
        CD-00002
      </ProductNumberByMerchant>
      <MerchantID type="">
        15430
      </MerchantID>
      <Quantity>
        23
      </Quantity>
      <UserData>
        <UserDataField name="attr1">
          "paul chik"
        </UserDataField>
      </UserData>
    </ProductInventoryInfo>
  </DataArea>
</Update_NC_ProductInventory>
```

```
</UserData>
  </ProductInventoryInfo>
</DataArea>
</Update_NC_ProductInventory>
```

The Update_NC_ProductInventory message would be put in the INBOUND message queue. We can check the status of the INBOUND queue by browsing the message queue. Refer to “Browse message” on page 415 for detailed instructions for browsing.

By default, the interval that the WebSphere Commerce Suite server will get messages from the queue is 30 seconds. This setting can be found in the ncei_ece.ini file with the following parameters:

- BACKGROUND_CYCLE_TIME 30
- BACKGROUND_INTER_JOB_TIME 30

Once, the message is processed by WebSphere Commerce Suite, the message will be removed from the INBOUND queue. The WebSphere Commerce server will update the inventory information entry of corresponding products within the PRODUCT database table on the WebSphere Commerce database server, based on the information in the XML message.

By default, the product page created by the store creator does not display inventory information. To add inventory display capability, we need to modify the net.data macro or JSP for the ProductDisplay.

17.4.1.4 Adding inventory information for ProductDisplay

In this example, we will provide instruction for modifying the net.data macro for ProductDisplay. The updated ProductDisplay will be visible only to registered users within the IBM shopper. Registered users of the Public shopper group should continue to see the default ProductDisplay page.

To modify the ProductDisplay macro, complete the following steps:

1. Copy the original proddisp.d2w to proddispinv.d2w file.
2. Add the proddispinv.d2w file to the ITSO studio project.
3. Modify proddispinv.d2w within the studio ITSO project:
 - a. Add the following net.data function within the function section of proddispinv.d2w:

```
%function(dtw_odbc) DISPLAY_PRODUCT_INVENT(){
    SELECT prvent
```

```

FROM product
WHERE pmenbr=$(pmenbr) and prrfnbr=$(prrfnbr)

%REPORT{

  %ROW{
    <BR><BR>
    <FONT size=$(textSize) FACE=$(textFont)>Inventory:
    $(V_prvent)
    </FONT><P>
  }

  %}

  %MESSAGE{100:{ %} :continue %}
%}

```

Insert the following line into the net.data report section of proddispinv.d2w within each section after the line containing @DISPLAY_PRODUCT_PRICE:

```
@DISPLAY_PRODUCT_INVENT()
```

- b. Save and close file.
4. Assign the template to the products by means of the WebSphere Commerce Administrator. We will show you how to do it for a single product. Repeat these steps for additional products.
 - a. Start WebSphere Commerce Administrator, log on as Administrator.
 - b. Click **Store Manager** -> **Product Information**.
 - c. From the Select Store drop down list, select **ITSO**. Key in the SKU number of the product or click the **Search** button to search for products information.
 - d. Click **Template** from the navigation bar.
 - e. You will see the Product Number or Item SKU box is filled by your SKU number. From the Shopper Group drop down list, select **IBM**.
 - f. At the Template box, key in the name of the net.data macro or JSP file name for the product display, and then click **Save**.

17.4.1.5 Back-end system simulation

For this example, we simulated a back-end systems by putting XML messages on the INBOUND queue. We manually created an XML document following the DTD format of the Update_NC_ProductInventory_10.dtd file. In addition, we wrote a program to put the XML document on the MQSeries

queue. For details refer to the *MQSeries Application Programming Guide*, SC33-0807-10.

The Document Type Definition (DTD) file defines the format of required XML messages. We used the IBM Visual XML tool to generate the XML document. The IBM XML Tools package includes a number of tools for creating, transforming, and querying XML documents. The tool can be downloaded from <http://www.alpha.works.ibm.com>. You can use any XML tool or a text editor if you are really careful typing.

In this example, we used the Update_NC_ProductInventory message. The inbound message contains inventory information for a product. The back-end application generates the message and puts it on the inbound message queue. Once the Commerce server receives the message, the PRODUCT table is updated with the new inventory information from the Commerce database. The Update_NC_ProductInventory message uses XML format from the Update_NC_ProductInventory_10.dtd file found at:

- Windows NT: c:\ibm\CommerceSuite\xml\MQSeriesAdapter
- AIX: /usr/lpp/CommerceSuite/xml/MQSeriesAdapter

17.5 MQSeries adapter debugging tips

If the MQSeries message queue is not receiving the order message from WebSphere Commerce, check the following:

- Ensure that the MQSeries client/server started successfully.
- Ensure that the MQSeries adapter started successfully.

The WebSphere Commerce Suite and MQSeries log files listed in Table 42 can be used to help debug the problem.

Table 42. WebSphere Commerce debug file locations

Product	File	Description
WebSphere Commerce	\ibm\CommerceSuite\instance\instance_name\logs*.log	Log files
	\ibm\CommerceSuite\instance\instance_name\config\nci_ece.ini	Configuration file

Product	File	Description
	\\ibm\CommerceSuite\instance\instance_name\config\ncommerce.ini	Configuration file
MQSeries	\\MQSeries\error*.*	Log and Dump files

In order to verify the MQSeries server, we can try to insert test message to all created queues from the Commerce server where the MQSeries client or MQSeries server are located.

To verify the MQSeries adapter, we need to verify the log files of WebSphere Commerce, ncommerceXXXXX.log and ncei_eceXXXXX.log file, by completing the following steps:

1. Make sure MQSeries is started and working properly.
2. We recommend that you change the log level of parameter MS_LOGLEVEL from 0 to 3 for ncei_ece.ini and ncommerce.ini files and restart the WebSphere Commerce server.
3. Check the ncommercexxxx.log file. In our case we have assigned the overridable function ECEExtXMLOrdProc, to the task named EXT_ORD_PROC for the outbound message. We should see an entry like the following in the log file:

```
20000605103431  DEBUG  CMN0115D: Registered 'ECEExtXMLOrdProc_1.000 (IBM,NC)'
```
4. If you do not see this entry, or there is entry for registration failure for the overridable function, verify the task assignment procedure for the overridable function. Make sure the same user specified for the ncmqbridge command is specified in the Site Administrators group.
5. Correct the error and restart the WebSphere Commerce server. Change the value of MS_LOGLEVEL back to 0, and restart the server.

Sometimes, we find that the XML generated from the back-end system cannot be used to update the relative information of the WebSphere Commerce database. Verify the functionality of the MQSeries server, queue, and outbound message. Next we will focus on the format of the XML message generated from the back-end system by completing the following steps:

1. Make sure the XML document complies with the definition from the DTD file provided by WebSphere Commerce Suite. We can verify the XML document by using parser software.

2. Remove the new line character on the XML message. Otherwise, the message will be separated into different messages in the MQSeries queue. To view the message queue, refer to “Browse message” on page 415.
3. Without putting double quotes around the text information, you will see an error messages found in the ncei_ecexxxxx.log file like the following message:

```

20000612173805 ERROR CMN8122E: Failed processing command: ParseMsg
20000612173805 ERROR CMN8117E: Failed receiving MQ message <?xml
version="1.0"?><!DOCTYPE Update_NC_ProductInventory SYSTEM
"e:\mq\Update_NC_ProductInventory_IO.dtd"><Update_NC_ProductInventory
version="1.0"><ControlArea><Verb value="Update">Update</Verb><Noun
value="NC_ProductInventory">Update product inv</Noun></ControlArea><DataArea><P
roductInventoryInfo><ProductNumberByMerchant>8635</ProductNumberByMerchant><Mer
chantID
type="15430</MerchantID><Quantity>9999</Quantity><UserData><UserDataField
name="attr1">paul
chik</UserDataField></UserData></ProductInventoryInfo></DataArea></Update_NC_Pr
oductInventory>.
20000612173905 ERROR CMN8106E: Failed processing quantity update message.
20000612173905 ERROR CMN0205E: Command 'ProductUpdateQuantity_1.000 (IBM,NC)'
failed CheckParameters.
20000612173905 ERROR CMN8122E: Failed processing command:
ProductUpdateQuantity
20000612173905 ERROR CMN0208E: Command 'ParseMsg_1.000 (IBM,NC)' failed
Process.

```

4. If the data is wrong, for example the merchant is not found, you will see an error message in the ncei_ecexxxxx.log file like the following:

```

20000612184910 ERROR CMN0007E: SQL statement failure 'SELECT
prspecial, prknutag, proid, prdconbr, pravdate, prrfnbr, prmenbr,
prbuyabl, prnbr, prpub, prvent, prstmp, prfield1, prfield2, prpsnbr,
prpgtype, prfield3, prfield4, prfield5, prpcode FROM product WHERE prnbr
= 'CD-00002' and prmenbr = '17116'.'.
20000612184910 ERROR CMN0005E: Database error code -1019: '[IBM] [CLI
Driver] [DB2/NT] SQL0206N "17116" is not a column in an inserted table,
updated table, or any table identified in a FROM clause or is not a valid
transition variable for the subject table of a trigger.
SQLSTATE=42703'.

```

Correct the error and import the XML message again.

5. When debugging is complete, be sure to change the value of MS_LOGLEVEL back to 0 in the ini files and restart the WebSphere Commerce Suite instance.

17.6 Where to find more information

- *IBM WebSphere Commerce Suite Pro Edition for AIX, MQSeries Adapter V4.1*

- *IBM MQSeries for AIX, Quick Beginnings V5.1*, GC33-1867-02
- *IBM MQSeries for NT, Quick Beginnings V5.1*, GC34-5389
- *MQSeries Clients*, GC33-1632-07
- *MQSeries Application Programming Guide*, SC33-0807-10
- General IBM MQSeries information:
<http://www.ibm.com/software/ts/mqseries/>.

Appendix A. Runtime environment - Windows NT platform

This section provides detailed instructions for the Windows NT platform on how to install and configure a remote DB2 database server, and WebSphere Commerce Suite Pro V4.1 server. We recommend using the working example runtime environment instructions in conjunction with the *IBM WebSphere Commerce Suite Pro Edition for NT, Installation Guide V4.1* for specific questions about your runtime environment.

This appendix is organized into the following sections:

- Remote DB2 database server installation
- WebSphere Commerce Suite installation
- WebSphere Commerce Suite configuration

A.1 Remote DB2 database server installation

This section provides detailed instructions for installing the remote DB2 database server for the Windows NT platform.

A.1.1 Pre-installation requirements

Prior to the installation of the remote DB2 database server, you must ensure that you meet the hardware and software requirements. Refer to Chapter 1 of the *IBM DB2 Universal Database for Windows NT, Quick Beginnings V6*, GC09-2835 for a detailed list of prerequisite hardware, and software. In addition, refer to the *IBM WebSphere Commerce Suite Pro Edition for NT, Installation Guide V4.1*.

A.1.1.1 Hardware

In the working example, we used an IBM Netfinity Server.

- IBM Netfinity 3000
 - 500 MHz PIII
 - 512 GB RAM
 - 9 GB DASD

A.1.1.2 Software

In the working example, we used Windows NT 4.0 Server, Windows NT 4 service pack 6a.

A.1.1.3 Other requirements

- Before installing, you must log on to the local Windows NT User Manager Database.
- If your Windows NT system is an additional server or backup domain controller, create a Windows NT user for DB2 as a member of the administrators group on the local Windows NT User Manager Database. For example:
 - Userid: wcsadmin
 - Password: <your_password>
- Log on to Windows NT as the user created in the previous step.
- Shut down any other programs so that the setup program can update the required files.

A.1.2 IBM DB2 UDB EE V6.1.0.6 installation

To install DB2 UDB Enterprise Edition V6.1.0.6, complete the following steps:

1. Insert the DB2 UDB V6.1.0.6 CD in the CD-ROM drive.
2. From the root directory of the CD, run setup.exe.
3. When the Welcome window appears, click **Next**.
4. When the Select Products window appears, select **DB2 Enterprise Edition**, then click **Next**.
5. When the Select Installation Type window appears, click **Typical**.
6. When the Choose Destination Location window appears, click **Browse**.
7. When the Choose Folder window appears, enter the following path:
 - C:\IBM\SQLLIB
 - Click **OK**
8. When the Do you want the folder to be created windows appears, click **Yes**.
9. Click **Next** to continue.
10. When the Enter Username and Password for the Administration Server window appears, accept the defaults:
 - User ID: db2admin
 - Password: <your_password>
 - Confirm password: <your_password>
 - Click **Next**.

11. When the Question window will appear with the message, The username db2admin does not exist, would you like Setup to create it for you?, click **Yes**.
12. When the Start Copying Files window appears, review or change any settings; if you agree with the setting click **Next** to begin copying files.
13. When the Setup complete window appears, select **Yes, I want to reboot my computer now**, click **Finish** to reboot.
14. After the reboot, the DB2 First Steps window will appear. We recommend that you create the sample database to verify your DB2 server.

A.2 WebSphere Commerce Suite installation

If you plan on using a remote database server configuration, we recommend that you follow our installation instructions in conjunction with the *IBM WebSphere Commerce Suite Pro Edition for NT, Installation Guide V4.1*.

For the working example, we did not use the WebSphere Commerce Suite setup.exe found in the root of the CD. The procedure documented in this section installs and verifies each component individually during the installation process. We believe that this procedure allows for greater flexibility and provides methods for verifying components before proceeding. As a result, this procedure may take slightly longer than the standard install.

The WebSphere Commerce server installation includes the following server components:

- Microsoft Windows NT 4.0 Server
- Microsoft Windows NT 4.0 service pack 6a
- IBM JDK V1.7
- IBM HTTP Server (Apache) V1.3.6.2
- IBM DB2 UDB V6.1.0.6 Client
- IBM WebSphere Application Server Advanced Edition V3.02
- IBM Net.Data V6.1
- IBM WebSphere Commerce Suite Pro V4.1

A.2.1 Pre-installation requirements

Prior to the installation you must ensure that you meet the hardware and software requirements. Refer to Chapter 1 of the *IBM WebSphere Commerce Suite Pro Edition for NT, Installation Guide V4.1*, for requirements.

A.2.1.1 Hardware

In the working example, we used an IBM Netfinity Server.

- IBM Netfinity 3000
 - 500 MHz PIII
 - 512 GB RAM
 - 9 GB DASD

A.2.1.2 Software

In the working example, we used Windows NT 4.0 Server, Windows NT 4 service pack 6a.

A.2.1.3 Other requirements

- Before installing, you must log on to the local Windows NT User Manager Database.
- If your Windows NT system is an additional server or backup domain controller, create a Windows NT user as a member of the administrators group on the local Windows NT User Manager Database. For example:
 - User ID: wcsadmin
 - Password: <your_password>
- Log on to Windows NT as the user created in the previous step.
- Shut down any other programs so that the setup program can update the required files.
- Install Netscape Communicator V4.61 on the Windows machine that you will use to access the Commerce Suite Administrator (provided in the root of the WebSphere Commerce Suite V4.1 CD).

A.2.2 IBM JDK 1.1.7

This section provides detailed instructions for installing the JDK on the WebSphere Commerce server system.

A.2.2.1 Install IBM JDK 1.1.7

To install the IBM JDK 1.1.7, complete the following steps:

1. Insert the WebSphere Application Server CD, where *e:* is the CD-ROM drive.
2. From the *e:\nt\jdk* directory of the CD, run *setup.exe*.
3. When the IBM Developer Kit, JDK 1.1.7 Welcome window appears, click **Next**.

4. When the Software License Agreement window appears, click **Yes** to accept.
5. When the Select Components window appears, accept the default component selections, and click **Browse** to change the destination directory.
6. When the Choose Directory window appears, enter the following path:
 - Destination directory: c:\ibm\jdk
 - Click **OK**.
7. When the message, do you want the directory to be created appears, click **Yes**.
8. Click **Next** to continue.
9. When the Start Copying Files window appears, click **Next**.
10. When the Setup Complete window appears, click **Finish**.

A.2.2.2 Configure the JDK Path

Update the Windows NT environment path for the JDK. Refer to the JDK readme for details.

1. Select **My Computer**, right click **Properties**, select **Environment**.
2. Select **Path**, add c:\ibm\jdk\bin to the path. Remember to separate entries by using the “;” .
3. Click **Set**, click **Apply**, click **OK**.

A.2.2.3 Verify IBM JDK 1.1.7

1. Click **Start -> Programs -> Command Prompt**.
2. From the command prompt, type the following:

```
c:\ java -fullversion
```
3. You should see the following message displayed:

```
java full version "JDK 1.1.7 IBM build n117p-19990823 (JIT enabled: ibmjitc)"
```
4. Exit the command prompt window by typing the following:

```
c:> exit
```

A.2.3 Install IBM HTTP Server V1.3.6.2 (Apache)

To install the IBM HTTP Server V1.3.6.2, complete the following steps:

A.2.3.1 Pre-installation requirements

Create a Windows NT user as a member of the administrators group on the local Windows NT User Manager Database. This user ID will be used for HTTP Server to run as a Windows NT service. The userid must exist prior to the installation of the HTTP Server. For example:

- User ID: httprun
- Password: <your_password>

A.2.3.2 Install HTTP Server

To install the HTTP Server, complete the following steps:

1. Insert the WebSphere Application Server CD, where *e:* is the CD-ROM drive.
2. From the **e:\nt\httpd** directory of the CD, run **setup.exe**.
3. When the Choose Setup Language window appears, select a language, click **OK**.
4. When the Welcome window appears, click **Next**.
5. When the Software License Agreement window appears, click **Yes** to accept.
6. When the Choose Destination Directory window appears, click **Browse**.
7. When the Choose folder window appears, type path **c:\ibm\http**, click **OK**.
8. When the message, do you want the folder to be created appears, click **Yes**.
9. Click **Next** to continue.
10. When the Setup Type window appears, select **Typical**, and then click **Next**.
11. When the Select Program Folder window appears, accept the default, and then click **Next**.
12. When the Information for Service Setup window appears. The HTTP Server will be installed as a service on Windows NT. Please enter the user ID and password that this service will logon and run under on Windows NT. This user ID must be valid on the local machine (created during the pre-installation step).
 - User ID: httprun
 - Password: <your_password>
 - Verify Password: <your_password>
 - Click **Install**.

13. When the Setup Complete Window appears, select **Yes, I want to restart my computer now**, and then click **Finish**.

A.2.3.3 Configure HTTP Server

To configure the IBM HTTP Server and verify that it is working properly, complete the following steps:

Create IBM HTTP Administration Server - admin user ID

The IBM HTTP Administration Server admin user ID is used to configure and manage the IBM HTTP Server.

1. **Start -> Programs -> Command Prompt**

2. From the command prompt type the following:

```
c:\> cd \ibm\http
c:\ibm\http> htpasswd -m conf\admin.passwd httpadm
- New password: <your_password>
- Re-type new password: <your_password>
```

3. Exit the command prompt by typing the following:

```
c:\ibm\http> exit
```

Enabling SSL for the HTTP Server

SSL is required by the WebSphere Commerce Suite. There are two methods that can be used to enable SSL:

- Production SSL Enablement - please refer to Chapter 11 in the *IBM WebSphere Commerce Suite Pro Edition for NT, Installation Guide V4.1*.
- Test SSL Enablement - for the purpose of the working example, we will create a security key file for testing.

To enable a SSL for test purposes, complete the following steps:

1. Stop the IBM HTTP Server:

- Click **Start -> Settings -> Control Panel -> Services**.
- Select **IBM HTTP Server**, click **Stop**.
- Select **IBM HTTP Administration Server**, click **Stop**.
- Close the **Services** dialog.

2. Click **Start -> Programs -> Command Prompt**.

3. From the command prompt, type the following:

```
c:\> cd \ibm\http\conf
```

```
c:\ibm\http\conf> copy httpd.conf httpd.conf.orig
c:\ibm\http\conf> copy httpd.conf.sample httpd.conf
```

4. Open httpd.conf with a text editor.
5. Uncomment the following lines, by removing the # sign:
 - #LoadModule ibm_ssl_module modules/IBMModuleSSL<encryption_level>.dll
(where the encryption_level is the appropriate level of your locale)

Note

Where the <encryption_level> is the appropriate level of your locale (for example, 128 for en_US).

```
-#Listen 443
-#<VirtualHost host.some_domain.com:443>
```

Note

You must substitute your fully qualified host name in this line.

```
-#SSLEnable
-#</VirtualHost>
-#SSLDisable
-#Keyfile "c:/ibm/http/keys/keyfile.kdb"
```

Note

Replace the word **keys** with **ssl**.

```
-#SSLV2Timeout 100
-#SSLV3Timeout 1000
```

6. Save your changes and close the httpd.conf file.

Create test SSL certificate

1. Click **Start** -> **Programs** -> **IBM HTTP Server** -> **Start Key Management Utility**.
2. When the IBM Key Management window appears, click the **Key Database File** from the menu bar, click **New**.
3. New window appears, enter the following:
 - Filename: keyfile.kdb

- Location: c:\ibm\http\ssl
 - Click **OK**.
4. When the Password Prompt window appears, enter the following:
 - Password: <your_password> (HTTP Server runtime user password)
 - Confirm password: <your_password>
 - Select checkbox **Set expiration time**
 - Days: <enter_desired_number_of_days>
 - Select checkbox **Stash the password to a file**.
 - Click **OK**.
 5. When the Information window appears, click **OK**.
 6. Click **Create** from the menu bar, select **New Self-Signed Certificate**.
 7. When the Create New Self-Signed Certificate window appears, enter the following required fields:
 - Key Label: <user_defined_name_for_key>
 - Organization: <your_organization_name>
 - Accept the defaults for the remaining fields, and then click **OK**.
 8. Close the **Key Management Utility**.
 9. Start IBM HTTP Server, IBM HTTP Administration Server services
 - **Start -> Settings -> Control Panel -> Services**
 - Select **IBM HTTP Server**, click **Start**
 - Select **IBM HTTP Administration Server**, click **Start**
 - Close the **Services** dialog.

Configure HTTP Server - ServerName

1. Start a Web browser, enter URL http://<localhost>.
2. A Welcome to the IBM HTTP Server page is displayed, click **Configure server**.
3. The IBM HTTP Administration Server login window is displayed, enter the following:
 - User Name: httpadm
 - Password: <your_password>
 - Click **OK**.
4. Double-click **Basic Settings** from the left-hand navigation bar.

5. Double-click **Core Settings**, enter the following:
 - Server name: <your_hostname.domain.com>
 - Scroll down, click **Submit** to save.
6. Close the Web browser.

A.2.3.4 Verify HTTP Server

To verify the HTTP Server, complete the following steps:

1. Start your Web browser, disable and clear all disk and memory cache, and disable all proxy servers.
2. To verify the HTTP Server, enter URL: http:\\<your_hostname>
3. To verify that the HTTP Server is SSL is enabled, enter URL: https:\\<your_hostname>.
4. Close the Web browser.

HTTP Server Verification

Do not proceed until your Web server is working properly with SSL enabled.

A.2.4 Install IBM DB2 UDB EE V6.1.0.6

This section provides detailed instructions for installing the DB2 Administration Client.

A.2.4.1 Install IBM DB2 V6.1.0.6 client

To install the DB2 client, complete the following steps:

1. Insert the IBM DB2 V6.1.0. 6 CD, where *e:* is the CD-ROM drive
2. From the root directory of the CD, run setup.exe.
3. When the Welcome window appears, click **Next**.
4. When the Select Products window appears, select **DB2 Administration Client**, and then click **Next**. Make sure the other options are deselected.
5. When the Select Installation Type window appears, select **Typical**.
6. Choose Destination Location window appears, click **Browse**.
7. When the Choose Folder window appears, type path **c:\ibm\sqllib**, click **OK**.
8. When the message, do you want the folder to be created appears, click **Yes**.

9. Click **Next** to continue.
10. If the Windows NT NetBEUI protocol is installed, the Configure NetBIOS window will appear. Deselect Configure NetBIOS, and then click **Next**.
11. When the Start Copying Files window appears, click **Next**.
12. When the Setup complete window appears, select **Yes, I want to reboot my computer now**, click **Finish** to reboot.

A.2.4.2 Configure DB2 Administration Client

This section details the configurations steps necessary after the DB2 client installation.

Create the WebSphere Application Server Repository database

On the remote DB2 database server create the WebSphere Application Server Repository database by completing the following steps:

1. Click **Start -> Programs -> DB2 for Windows NT -> Command Window**.
2. From the DB2 command window, type the following:
c:\ibm\sqllib\bin> db2 create db wasnt
c:\ibm\sqllib\bin> db2 update db cfg for wasnt using applheapsz 512

Note: applheapsz

Increase the applheapsz to 512 to improve performance

Catalog the database node

On the WebSphere Commerce Server, complete the following steps:

1. Click **Start -> Programs -> DB2 for Windows NT -> Command Window**.
2. From the DB2 command window, type the following:
c:\ibm\sqllib\bin> db2 catalog tcpip node dbrsvnt remote dbrsvnt server 50000

Syntax

```
db2 catalog tcpip node <node_name> remote <db_server_hostname>  
server 50000
```

Verify attach to database node

On the WebSphere Commerce Server, complete the following steps:

1. Click **Start -> Programs -> DB2 for Windows NT -> Command Window**.

2. From the DB2 command window, type the following:

```
c:\ibm\sqllib\bin> db2 attach to dbsrvnt user db2admin using db2admin  
Instance Attachment Information  
Instance server      = DB2/6000 6.1.0  
Authorization ID    = DB2ADMIN  
Local instance alias = DBSRVNT
```

Syntax

```
db2 attach to <node_name> user <db2inst_ID> using <db2inst_password>
```

Catalog the database

On the WebSphere Commerce server catalog the WebSphere Application Server database by doing the following:

1. Click **Start -> Programs -> DB2 for Windows NT -> Command Window**.
2. From the DB2 command window, type the following:

```
c:\ibm\sqllib\bin> db2 catalog db wasnt at node dbsrvnt
```

Syntax

```
db2 catalog db <database_name> at node <node_name>
```

Verify connect to database

On the WebSphere Commerce server verify the connection to the database on the remote DB2 database server by doing the following:

1. Click **Start -> Programs -> DB2 for Windows NT -> Command Window**.
2. From the DB2 command window, type the following:

```
c:\ibm\sqllib\bin> db2 connect to wasnt user db2admin using db2admin  
Database Connection Information  
Database server      = DB2/6000 6.1.0  
SQL authorization ID = DB2ADMIN  
Local database alias = WASNT
```

Syntax

```
db2 connect to <db_name> user <db2inst_ID> using  
<db2inst_password>
```

A.2.5 Install WebSphere Application Server

This section provides detailed instructions for installing the WebSphere Application Server Advanced Edition V3.02.

On the WebSphere Commerce server, complete the following steps:

A.2.5.1 Installation Pre-requisites

- Stop the IBM HTTP Server and other Web servers
 - Click **Start** -> **Settings** -> **Control Panel** -> **Services**.
 - Select **IBM HTTP Administration Server**, click **Stop**.
 - Select **IBM HTTP Server**, click **Stop**.
 - Close the **Services** dialog
- You will need 30-50 MB free in your temp directory (usually on the C drive), even if you are installing on another drive. The Install shield package unpacks to the temp directory.
- Exit all Windows programs prior to the install.

A.2.5.2 Installing WebSphere Application Server

To install the WebSphere Application Server, complete the following steps:

1. Insert the WebSphere Application Server CD, where *e:* is the CD-ROM drive.
2. From the *e:\nt* directory of the CD, run *setup.exe*.
3. When the Choose Setup Language window appears, select language, click **OK**.
4. When the WebSphere Application Server window appears, click **Next**.
5. Install Options window appears:
 - Select **Custom Installation**
 - Select the destination directory field, type path: **c:\ibm\was**
 - Click **Next**.
6. When the Choose Application Server Components window appears:
 - Select **Configure Administrative domain with default application server and a default application**
 - Select plug-in **IBM HTTP Server V1.3.6**
 - Accept the defaults for the options already selected, refer to Figure 89 to see the selected components, click **Next**.

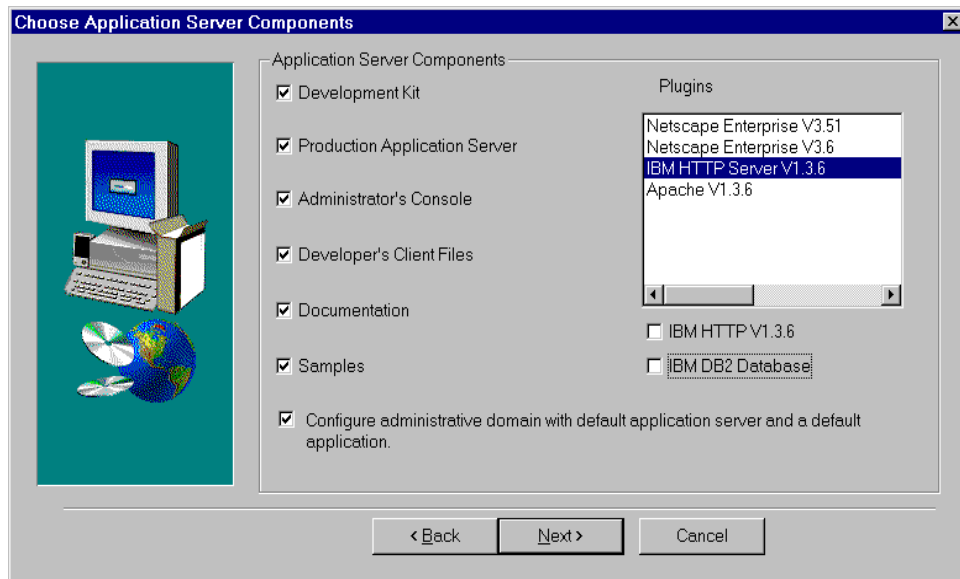


Figure 89. WebSphere Application Server Install - Components

Note

The checkbox for the IBM HTTP v 1.3.6 and IBM DB2 Database are not required if you have already installed these components prior to the WebSphere Application Server install as we have done.

7. When the Select Java Development Kit window appears, refer to Figure 90 for selections, and then click **Next**.

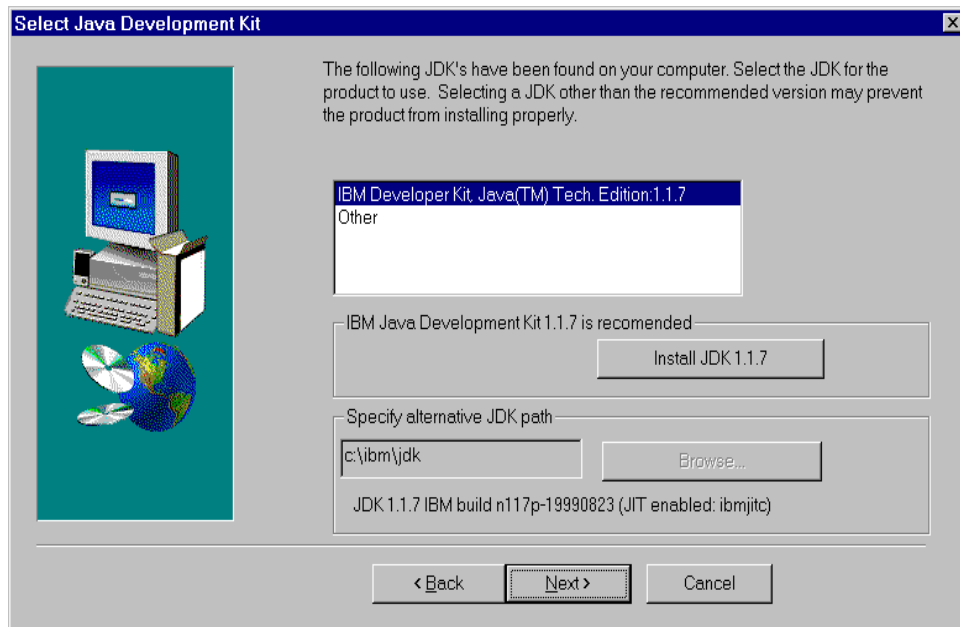


Figure 90. WebSphere Application Server Install - Java Development Kit

8. When the Security/Database Options window appears, refer to Figure 91 for entering and selecting the required fields, and click **Next**.

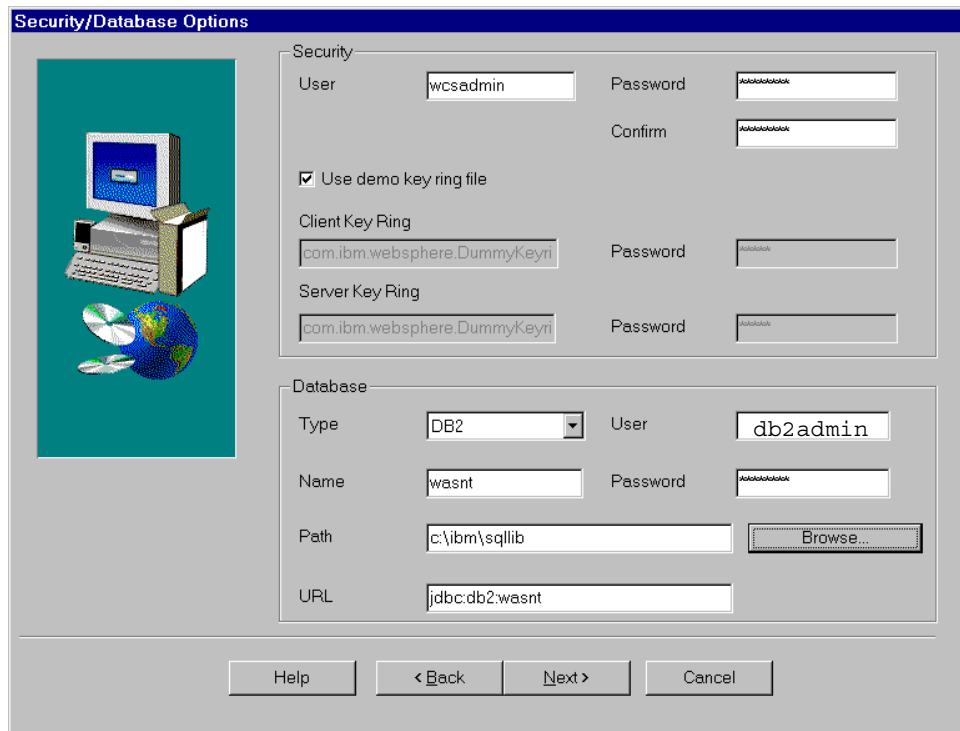


Figure 91. WebSphere Application Server Install - Security/Database Options

9. When the Select Program Folder window appears, accept the default, and then click **Next** to begin copying files.
10. When the Configure IBM HTTP Server: httpd.conf window appears, accept the path: **c:\ibm\http\conf**, and then click **OK**.
11. When the Setup Complete window appears, click **Finish**.
12. When the Restarting Windows window appears, select **No, I will restart my computer later**, and then click **OK**.
13. Since we have already created the WebSphere Application Server Repository database manually on the remote DB2 database server, we will rename the createwasdb.scr used by createdb2.bat that would normally run after the system reboot. If you always want to manually create your database, then rename createdb2.bat.
 - a. Click **Start -> Programs -> Command Prompt**.
 - b. From the command prompt, type the following:

```
c:\> cd \ibm\was\bin
```



```
c:\ibm\was\bin> ren createwasdb.scr createwasdb.scr.bak
c:\ibm\was\bin> exit
```

14. Shutdown and restart the system.

A.2.5.3 Start WebSphere Application Server

On the WebSphere Commerce server do the following:

1. After the system reboot, start the WebSphere AdminServer.
 - Click **Start -> Settings -> Control Panel -> Services**.
 - Select **IBM WS AdminServer**, double click, change startup type to **Automatic**, click **OK**.
 - Select **IBM WS AdminServer**, click **Start**.
 - Close the services dialog.
2. View the c:\ibm\was\logs\tracefile, it should look as follows:

```
008.212 184fad AdminServer    A Initializing WebSphere Administration server
008.242 184efd DrAdminServer  A DrAdmin available on port 1,049
058.274 184fad SASConfig      A
SAS.Property:com.ibm.CORBA.principalName.has.been.updated
062.100 184fad InitialSetupI  A Creating.Sample.Server.Configuration
088.938 184fad AdminServer    A WebSphere Administration server open for
e-business
```

A.2.5.4 Verify WebSphere Application Server

To verify that the WebSphere Application Server is running properly, complete the following steps:

1. Start WebSphere Administrators Console
 - Click **Start -> Programs -> IBM WebSphere -> Application Server V3.0 -> Administrators Console**.
2. When the WebSphere Advanced Administrative Console window appears, select the **Topology** tab in the left-hand console frame.
3. Expand **WebSphere Administrative Domain** and select **default_host**.
4. Select the **Advanced** tab in the right-hand pane.
5. Under Host Alias, add fully qualified hostname of your server. If you added your ServerName in the httpd.conf prior to the WebSphere Application Server install, this entry will already exist (Windows NT).
6. For each entry under Host Alias you must enter the same entry with the secure server port number appended. For example:
 - <hostname.some_domain.com>:443, click **Apply**

- <hostname>:443, click **Apply**
 - <ip_address>:443, click **Apply**
7. Expand <your_hostname>
 8. Under your hostname, select **Default Server**, click the green button on the console toolbar to start the Default Server.
 9. When the window appears stating that the “Default Server.start” completed successfully, click **OK**.
 10. Close the WebSphere Administrators Console.
 11. From a Web browser, enter URL: `https://<hostname>/servlet/snoop`
 12. You should see information about the following:
 - Requested URL
 - Init parameters
 - Request information
 - Request headers
 - Client cookies
 - ServletContext attributes
 13. Now that you have verified that the WebSphere Application Server is working properly you are ready to continue to the next step.

A.2.6 Install IBM Net.Data V6.1

This section provides detailed instructions for installing Net.Data on the WebSphere Commerce server system.

To install the Net.Data V6.1, complete the following steps:

1. Insert the IBM Net.Data V6.1 CD, where *e:* is the CD-ROM drive.
2. From the `e:\code\nt` directory of the CD, run `netdatantv61.exe`.
3. When the Choose Setup Language window appears, select a language, click **OK**.
4. When the Do you want to view the readme me now window appears, click **No**.
5. When the Welcome window appears, click **Next**.
6. When the Choose Destination Directory window appears, click **Browse**.
7. When the Choose folder window appears, type path `c:\ibm\netdata`, click **OK**.

8. When the message, do you want the folder to be created appears, click **Yes**.
9. Click **Next** to continue.
10. When the Net.Data Macro Directory window appears, click **Next**.
11. When the Information window appears, click **OK**.
12. When the Web server CGI-BIN directory window appears, click **Next**.
13. When the Web server HTML directory window appears, click **Next**.
14. When the Setup is complete window appears, select **No, I will restart my computer later**, click **OK**.

A.2.7 Install WebSphere Commerce Suite

This section provides detailed instructions for installing the WebSphere Commerce Suite.

A.2.7.1 Installation Pre-requisites

- Ensure that you have installed the IBM HTTP Server, DB2, Net.Data, JDK, and WebSphere Application Server before you install WebSphere Commerce Suite V4.1.
- Ensure that the IBM HTTP Server is started
 - Click **Start** -> **Settings** -> **Control Panel** -> **Services**.
 - Select **IBM HTTP Server**, click **Start**.
- Ensure that the IBM WS AdminServer is started
 - Click **Start** -> **Settings** -> **Control Panel** -> **Services**.
 - Select **IBM WS AdminServer**, click **Start**.
- Create a user for WCS - Windows NT server.

Create a Windows NT user as a member of the administrators group on the local Windows NT User Manager Database. This user ID will be used for WebSphere Commerce Server to run as a Windows NT service. The user ID must exist prior to the installation of WebSphere Commerce Suite. For example:

- User ID: wcsrwn
- Password: <your_password>

A.2.7.2 Installing WebSphere Commerce Suite

To install the WebSphere Commerce Suite, complete the following steps:

1. Insert the WebSphere Commerce Suite CD, where *e:* is the CD-ROM drive.
2. From the root directory of the CD, run setup.exe.
3. When the Welcome window appears, click **Next**.
4. When the Software License Agreement window appears, click **Accept**.
5. When the User ID and Password window appears, enter the following:
The WebSphere Commerce runtime user is used to log in and run as a service to Windows NT. The user must be valid with administrator authority on the local machine.
 - User ID: wcsr_un
 - Password: <your_password>
 - Password Verification: <your_password>

Note

This window will not be displayed if you installed the WebSphere Application Server prior to the WebSphere Commerce Suite, as we have done in this example.

6. When the Component Selection window appears, refer to Figure 92 for entering the following:
 - Select **Use Remote Database**.
 - Accept defaults for the remaining selections, click **Next**.

Note

Notice that the WebSphere Commerce Suite install has detected that the Web server, IBM HTTP Server 1.3.6, and the IBM Universal Database 6.1.0.6, have already been installed as shown in Figure 92.

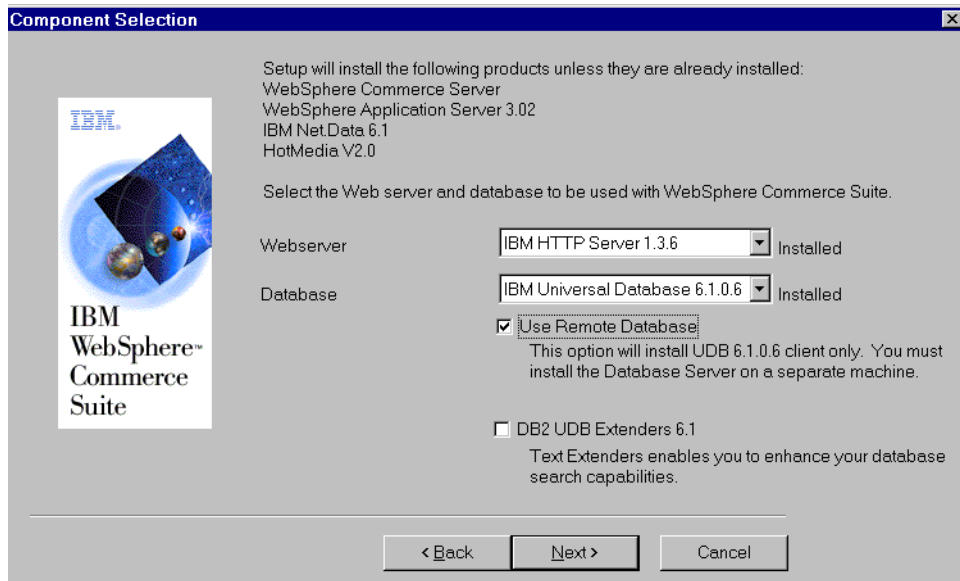


Figure 92. WebSphere Commerce Suite Install - Component Selection

7. When the Choose Destination window appears, type path: **c:\ibm\wcs**, click **Next**.
8. When Select Program Folder window appears, accept default, click **Next**.
9. When the Summary window appears, click **Next**.
10. When the IBM Product Registration window appears, enter data to register, click **Next**.
11. When the Installation Complete window appears, select **Yes, I want restart my computer now**, click **Finish**.
12. After the system restarts the Configuration Manager will be launched. Ignore this dialog for the time being. We need to configure some items before we enter the Configuration Manager to create our WebSphere Commerce instance. Continue to, A.3, "WebSphere Commerce Suite configuration" on page 443.

A.3 WebSphere Commerce Suite configuration

This section provides detailed instructions for configuring the WebSphere Commerce Suite.

A.3.1 Configuration Pre-requisites

After your WebSphere Commerce server restarts, the Configuration Manager will be launched. Before logging on, perform the following pre-requisite steps.

A.3.1.1 Copy netcpwd.dll to remote DB2 server

Copy the netcpwd.dll from c:\ibm\wcs\nc_schema\db2\udf\nt on your WebSphere Commerce server to c:\ibm\sqllib\function on your remote DB2 server.

A.3.1.2 Verify Services

Verify that the following services are started:

- IBM HTTP Server
- IBM DB2
- IBM WS AdminServer

A.3.2 Configuring WebSphere Commerce Suite components

This section describes how to configure the WebSphere Commerce Suite components. On the WebSphere Commerce server system, do the following:

1. After the system reboots, the WebSphere Commerce Configuration Manager is launched. Enter the following:
 - User ID: webadmin (default)
 - Password: webibm (default)
 - Click **OK**; the first time you log in you will be prompted to change your password.

Note

To start the WebSphere Configuration Manager:

Click **Start -> Programs -> IBM WebSphere Commerce Suite -> Configuration**.

2. The Configuration Manager window appears, click **New** to create a new WebSphere Commerce Instance.
3. Select the **Database** tab
4. In the Database tab, as seen in Figure 93, enter the following:
 - Database Name: dbwcsnt

- DBMS: IBM Universal Database
- Instance Owner ID: db2admin
- Database User Logon: db2admin
- Database Logon Password: <your_password>
- Database Option: (ignore this option for the working example)
- Use Remote Database: **enable checkbox**
- Remote Database Location: dbsrvnt (remote database server hostname)
- Remote Database Node Name: dbsrvnt (as defined A.2.4.2, "Configure DB2 Administration Client" on page 433)

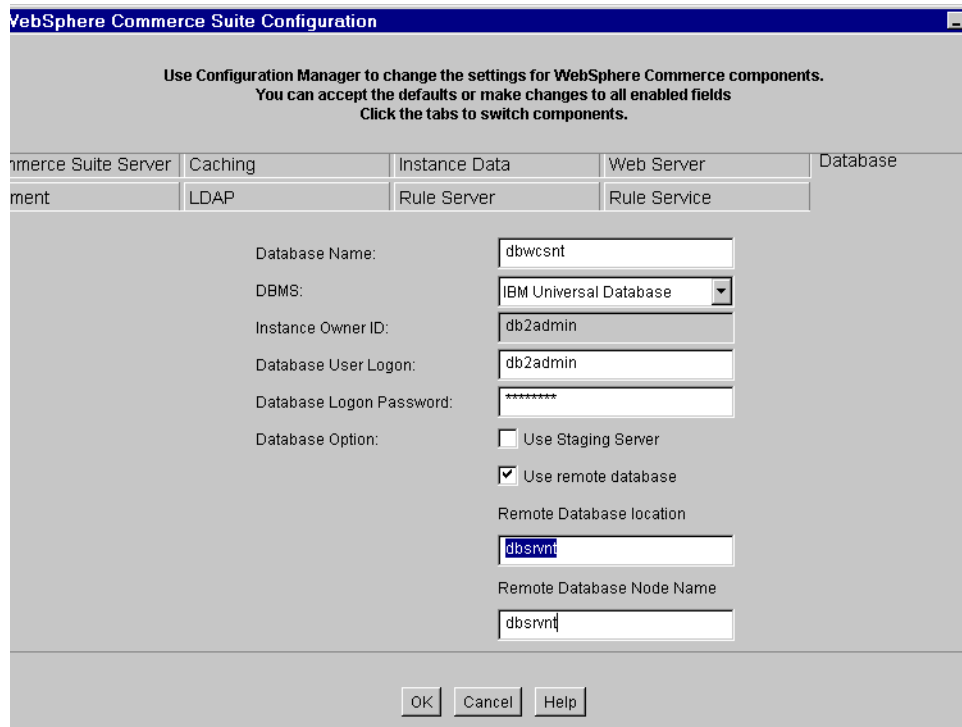


Figure 93. WebSphere Configuration Manager - database tab

5. Select the **Commerce Suite Server** tab.
6. In the Commerce Suite Server tab, as seen in Figure 94, enter the following:
 - Instance Name: wcsnt

- Leave the remaining settings as default.
- Click **OK** to create the WCS instance.

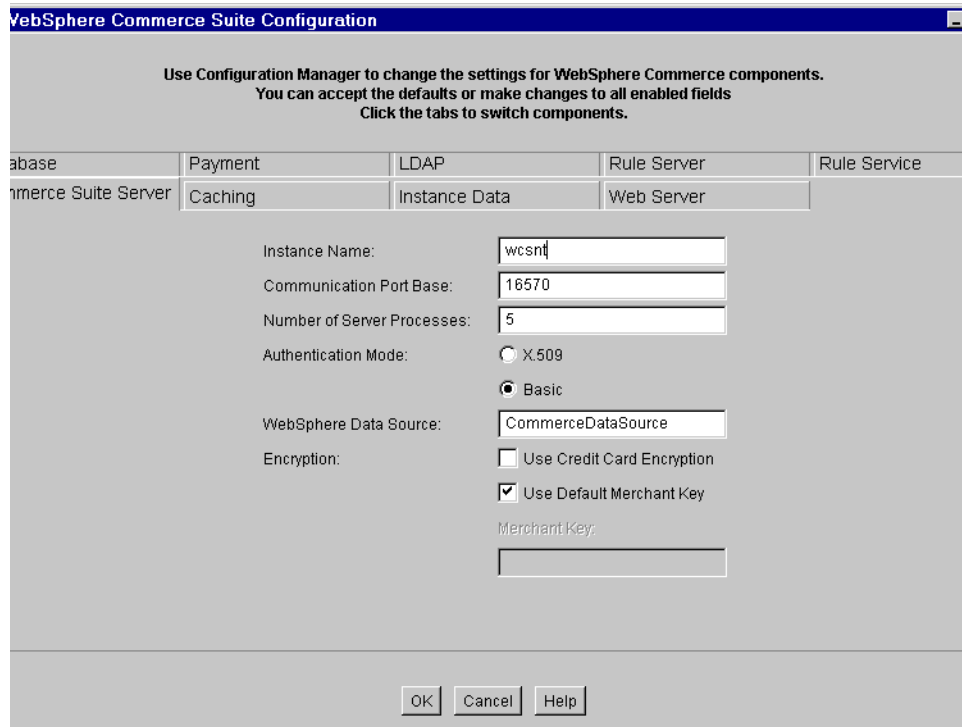


Figure 94. WebSphere Configuration Manager- Commerce Suite tab

7. Press **Start** to start the WCS instance.
8. Click **Exit**, to close the Configuration Manager.

A.3.2.1 Post Configuration

The following steps must be completed after the WebSphere Commerce Suite configuration.

Start WebSphere Commerce Server

1. Click **Start** -> **Programs** -> **IBM WebSphere** -> **Application Server V3.0** -> **Administrator's Console**.
2. When the WebSphere Advanced Administrative Console window appears.
3. When the Select the **Topology** tab in the left-hand console frame.
4. Expand **WebSphere Administrative Domain**.

5. Select and expand <your_hostname>.
6. Select **WebSphere Commerce Server** on the left-hand console frame.
7. Click the green button on the console toolbar to start the WebSphere Commerce Server.
8. When the Information dialog window appears, it should say:
Command “WebSphere Commerce Server.start” completed successfully
 - Click **OK**.

Start WebSphere Commerce Suite - ncdadmin

1. Start the Web browser (Netscape 4.61 or higher), enter URL:
http://<your_hostname>/ncadmin
2. WebSphere Commerce Suite - ncdadmin window appears, enter the following:
 - User Name: ncdadmin (default user)
 - Password: ncdadmin (default password)
For security reasons, please change your password to something other than ncdadmin.
 - New Password: <your_password>
 - Verify New Password: <your_password>
3. The WebSphere Commerce Suite Administration window should be displayed. We explain how to use this tool in greater detail in Chapter 13, “Create and publish a store” on page 273, and Chapter 15, “Java technologies in WebSphere Commerce Suite” on page 327.

Appendix B. SecureWay Directory (LDAP) - Windows NT platform

In this appendix we have included some special instructions for installing the SecureWay Directory product on the Windows NT platform. We have found that the SecureWay Directory V3.1.1.2 for Windows NT, does not work properly with DB2 UDB V6.1.0.6 for Windows NT (equivalent of V6.1 + fixpak 2), which is shipped with the WebSphere Commerce Suite product. It does work with DB2 V5.2 + fixpak 11. To correct this problem, we used SecureWay Directory V3.1.1.5 with DB2 UDB 6.1.0.6 (equivalent of V6.1 + fixpak 2).

For detailed instructions on installing the SecureWay Directory (LDAP) V3.1.1.5 for Windows NT product, refer to the product documentation. In addition, refer to Chapter 14, "SecureWay Directory (LDAP) - AIX platform" on page 289 for WebSphere Commerce Suite information. Once the server is installed and configured, the platform specific issues are minimal.

The IBM SecureWay Directory V3.1.1.5 for Windows NT product and documentation can be downloaded from:
<http://www.ibm.com/software/network/directory/downloads/>.

Note

The online documentation can be accessed after installation using a Web browser as follows, where x:\<inst_dir> is the root installation directory (default: C:\Program Files\IBM\LDAP\), and <language> is the installation language directory (default: enUS1252 for U.S. English):

- The *Installation and Configuration Guide*:
x:\<inst_dir>\nls\html\<language>\config\wparent.htm
- The *Administration Help*:
x:\<inst_dir>\web\<language>\help\parent.htm
- The *Directory Management Tool*:
x:\<inst_dir>\nls\html\<language>\dmt\dparent.htm
- The *C Programming Reference*:
x:\<inst_dir>\doc\progref.htm
- The *JNDI Programming Guide*:
Unzip x:\<inst_dir>\java\ibmjndi.zip and load ibmjndi\Guide.html

Appendix C. Special notices

This publication is intended to help I/T architects, I/T specialists, and developers in the design, development, and deployment of e-business applications. The information in this publication is not intended as the specification of any programming interfaces that are provided by WebSphere Commerce Suite V4.1, Pro Edition, or WebSphere Application Server V3.02, Advanced Edition. See the PUBLICATIONS section of the IBM Programming Announcement for the IBM WebSphere Commerce Suite V4.1, Pro Edition, and the IBM WebSphere Application Server V 3.02, Advanced Edition, for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM ("vendor") products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this

information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.


Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

Any performance data contained in this document was determined in a controlled environment, and therefore, the results that may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

This document contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples contain the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

AIX	AS/400
CICS	DB2
IBM	MQSeries
Nways	OS/390
OS/400	RS/6000
Redbooks	Redbooks Logo 
S/390	SecureWay
SP	TeamConnection
VisualAge	WebSphere
Wizard	Lotus Domino

The following terms are trademarks of other companies:

Tivoli, Manage. Anything. Anywhere., The Power To Manage., Anything. Anywhere., TME, NetView, Cross-Site, Tivoli Ready, Tivoli Certified, Planet Tivoli, and Tivoli Enterprise are trademarks or registered trademarks of Tivoli Systems Inc., an IBM company, in the United States, other countries, or both. In Denmark, Tivoli is a trademark licensed from Kjøbenhavns Sommer - Tivoli A/S.

C-bus is a trademark of Corollary, Inc. in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

PC Direct is a trademark of Ziff Communications Company in the United States and/or other countries and is used by IBM Corporation under license.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through The Open Group.

SET and the SET logo are trademarks owned by SET Secure Electronic Transaction LLC.

Other company, product, and service names may be trademarks or service marks of others.

Appendix D. Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

D.1 IBM Redbooks

For information on ordering these publications see “How to get IBM Redbooks” on page 461.

- *Servlet and JSP Programming with IBM WebSphere Studio and VisualAge for Java*, SG24-5755
- *Application Server Solution Guide, Enterprise Edition: Getting Started*, SG24-5320
- *WebSphere V3 Performance Tuning Guide*, SG24-5657
- *WebSphere Scalability: WLM and Clustering Using WebSphere Application Server Advanced Edition*, SG24-6153
- *Patterns for e-business: User-to-Business Patterns for Topology 1 and 2 using WebSphere Advanced Edition*, SG24-5864
- *IBM WebSphere and VisualAge for Java Database Integration with DB2, Oracle, and SQL Server*, SG24-5471
- *IBM WebSphere Performance Pack: Load Balancing with IBM SecureWay Network Dispatcher*, SG24-5858
- *Design and Implement Servlets, JSPs, and EJBs for IBM WebSphere Application Server*, SG24-5754
- *A Secure Way to Protect Your Network: IBM SecureWay Firewall for AIX V4.1*, SG24-5855
- *Understanding LDAP*, SG24-4986
- *WWW Programming: VisualAge for C++ and ST*, SG24-4734
- *Developing an e-business Application for the IBM WebSphere Application Server*, SG24-5423
- *LDAP Implementation Cookbook*, SG24-5110

D.2 IBM Redbooks collections

Redbooks are also available on the following CD-ROMs. Click the CD-ROMs button at ibm.com/redbooks for information about all the CD-ROMs offered, updates and formats.

CD-ROM Title	Collection Kit Number
IBM System/390 Redbooks Collection	SK2T-2177
IBM Networking Redbooks Collection	SK2T-6022
IBM Transaction Processing and Data Management Redbooks Collection	SK2T-8038
IBM Lotus Redbooks Collection	SK2T-8039
Tivoli Redbooks Collection	SK2T-8044
IBM AS/400 Redbooks Collection	SK2T-2849
IBM Netfinity Hardware and Software Redbooks Collection	SK2T-8046
IBM RS/6000 Redbooks Collection	SK2T-8043
IBM Application Development Redbooks Collection	SK2T-8037
IBM Enterprise Storage and Systems Management Solutions	SK3T-3694

D.3 Other resources

These publications are also relevant as further information sources:

- John Barry et al, *Developing Object-oriented Software - An Experienced-Based Approach*, Prentice Hall, 1997, ISBN 0137372485
- E. Gamma, R. Helm, R. Johnson, J. Vlissides, *Design Patterns - Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1995, ISBN 0201633612
- F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, M. Stahl, *Pattern-Oriented Software Architecture - A System of Patterns*, Wiley, 1996, ISBN 0471958697
- Coplien, J., *Advanced C++ Programming Styles and Idioms*, Addison-Wesley, 1991, ISBN 0201548550
- P. Monday, J. Carey, M. Dangler, *San Francisco Component Framework: An Introduction*, Addison-Wesley, 1999, ISBN 0201615878
- C. Alexander, S. Ishikawa, M. Silverstein, M. Jacobson, I. Fiksdahl-King, S. Angel, *A Pattern Language*. Oxford University Press, 1977, ISBN 0195019199
- J. Vlissides, *Pattern Hatching - Design Patterns Applied*, Addison Wesley, 1998, ISBN 0201432935
- "Enterprise Solutions Structure" in *IBM Systems Journal*, Volume 38, No. 1, 1999, available at: <http://www.research.ibm.com/journal/sj38-1.html>

- L. Bass, P. Clements, R. Kazman, *Software Architecture in Practice*, Addison Wesley, 1998, ISBN 0201199300
- Booch, Grady, *Object-Oriented Analysis and Design with Applications* (Addison-Wesley Object Technology Series), Addison-Wesley, 1994, ISBN 0805353402
- Jacobson, Ivar, *Object-Oriented Software Engineering; A Use Case Driven Approach*, Addison-Wesley, 1992, ISBN 0201544350
- Rumbaugh, James et al., *Object-Oriented Modeling and Design*, Prentice Hall, 1991, ISBN 0136298419
- Fowler, Martin, Kendall Scott (Contributor) and Ivar Jacobson, *UML Distilled; Applying the Standard Object Modeling Language*, Addison-Wesley, 1997, ISBN 0201325632
- *Designing e-business Solutions for Performance*, white paper by Maggie Archibald and Mike Schlosser, available at:
<http://www.ibm.com/software/developer/library/patterns/performance.html>
- JavaSoft: “*The Java HotSpot Performance Engine Architecture*” white paper, available at: <http://java.sun.com/products/hotspot/whitepaper.html>
- *IBM Application Framework for e-business*: white papers available at:
<http://www.ibm.com/software/ebusiness/>
- Flanagan, David, *JavaScript: The Definitive Guide*, Third Edition, O'Reilly & Associates, Inc., 1998, ISBN 1565923928
- Maruyama, Hiroshi, Kent Tamura and Naohiko Uramoto, *XML and Java: Developing Web Applications*, Addison-Wesley, 1999, ISBN 0201485435
- Flanagan, David, Jim Farley, William Crawford and Kris Magnusson, *Java Enterprise in a Nutshell*, O'Reilly & Associates, Inc. 1999, ISBN 1565924835
- Booch, Grady, *Object-Oriented Analysis and Design with Applications* (Addison-Wesley Object Technology Series), Addison-Wesley, 1994, ISBN 0805353402
- Jacobson, Ivar, *Object-Oriented Software Engineering; A Use Case Driven Approach*, Addison-Wesley, 1992, ISBN 0201544350
- Rumbaugh, James et al, *Object-Oriented Modeling and Design*, Prentice Hall, 1991, ISBN 0136298419
- Nagaratnam, Nataraj et al. 2000. *Security Overview of IBM WebSphere Standard/Advance 3.02*, IBM white paper, available at:
<http://www.ibm.com/software/webservers/appserv/whitepapers.html>

- *Developing Dynamic Web Sites Using the WebSphere Application Server* by Shane Claussen and Mike Conner, available at:
<http://service2.boulder.ibm.com/devcon/news0399/artpage2.htm>
- *IBM DB2 Universal Database for UNIX, Quick Beginnings V6*, GC09-2836
- *IBM DB2 Universal Database for Windows NT, Quick Beginnings V6*, GC09-2835
- *IBM MQSeries for AIX, Quick Beginnings V5.1*, GC33-1867-02 found at:
<http://www.ibm.com/software/ts/mqseries/library/manualsa/>.
- *IBM MQSeries for NT, Quick Beginnings V5.1*, GC34-5389 found at:
<http://www.ibm.com/software/ts/mqseries/library/manualsa/>.
- *MQSeries Clients*, GC33-1632-07 found at:
<http://www.ibm.com/software/ts/mqseries/library/manualsa/>.
- *MQSeries Application Programming Guide*, SC33-0807-10 found at:
<http://www.ibm.com/software/ts/mqseries/library/manualsa/>.
- *IBM WebSphere Commerce Suite Fundamentals*, GC09-2994
- *IBM WebSphere Commerce Suite Pro Edition for AIX v 4.1 Installation Guide*, packaged with the WebSphere Commerce Suite V4.1 product
- *IBM WebSphere Commerce Suite Pro Edition for NT Installation Guide V4.1*, packaged with the WebSphere Commerce Suite V4.1 product
- *IBM WebSphere Commerce Suite Pro Edition, Commands, Task, Overridable Functions and Database Tables V4.1*, packaged with the WebSphere Commerce Suite V4.1 product
- *Net.Data Administration and Programming Guide for OS/2, Windows NT, and UNIX* found at: <http://www.software.ibm.com/data/net.data/>.

D.4 Referenced Web sites

These Web sites are also relevant as further information sources:

- <http://www.ibm.com/software/developer/web/patterns>. IBM Patterns for e-business site.
- <http://www.ibm.com/software/webservers/appserv/> IBM WebSphere Application Server
- <http://www.as400.ibm.com/WebSphere> IBM WebSphere for AS/400
- <http://www.s390.ibm.com/oe/perform/dgwperf.html> Performance information for the Domino Go Webserver for OS/390

- <http://www.ibm.com/software/ebusiness> IBM's Application Framework for e-business
- <http://www.ecma.ch/stand/ECMA-262.htm> Standard ECMA-262 ECMAScript Language Specification
- <http://www.ibm.com/developer/xml> IBM XML information
- <http://www.javasoft.com/products/xml> SUN XML information
- <http://www.javasoft.com/products> Java product and API information
- <http://www.javasoft.com/products/jsp> JavaServer Page information
- <http://www.javasoft.com/products/servlet> Java Servlet information
- <http://www.w3.org/MarkUp> W3C's home page for HTML
- <http://oss.software.ibm.com/developerworks/opensource/jsp/index.html> JSP Format Bean Library
- <http://www.omg.org> Object Management Group (OMG) home page
- <http://www.rational.com/products/rose/> Rational Rose product information
- <http://www.ibm.com/software/vadd> IBM VisualAge Developer Domain
- (<http://www.cert.org> CERT home page)
- <http://www.as400.ibm.com/developer/java/deploy/deployguide.html> Information on deploying Java applications in an AS/400 environment
- <http://www.ibm.com/software/secureway/> IBM SecureWay products
- <http://www.ibm.com/software/network/directory/downloads/> IBM SecureWay Directory downloads
- <http://www.ibm.com/software/webservers/commerce/community/> IBM WebSphere Commerce Community
- <http://www.ibm.com/developer/features/framework/framework.html> IBM Application Framework for e-business: Web Application Programming Model

How to get IBM Redbooks

This section explains how both customers and IBM employees can find out about IBM Redbooks, redpieces, and CD-ROMs. A form for ordering books and CD-ROMs by fax or e-mail is also provided.

- **Redbooks Web Site** ibm.com/redbooks

Search for, view, download, or order hardcopy/CD-ROM Redbooks from the Redbooks Web site. Also read redpieces and download additional materials (code samples or diskette/CD-ROM images) from this Redbooks site.

Redpieces are Redbooks in progress; not all Redbooks become redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

- **E-mail Orders**

Send orders by e-mail including information from the IBM Redbooks fax order form to:

	e-mail address
In United States or Canada	pubscan@us.ibm.com
Outside North America	Contact information is in the "How to Order" section at this site: http://www.elink.ibm.com/pbl/pbl

- **Telephone Orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	Country coordinator phone number is in the "How to Order" section at this site: http://www.elink.ibm.com/pbl/pbl

- **Fax Orders**

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	Fax phone number is in the "How to Order" section at this site: http://www.elink.ibm.com/pbl/pbl

This information was current at the time of publication, but is continually subject to change. The latest information may be found at the Redbooks Web site.

IBM Intranet for Employees

IBM employees may register for information on workshops, residencies, and Redbooks by accessing the IBM Intranet Web site at <http://w3.itso.ibm.com/> and clicking the ITSO Mailing List button. Look in the Materials repository for workshops, presentations, papers, and Web pages developed and written by the ITSO technical professionals; click the Additional Materials button. Employees may access MyNews at <http://w3.ibm.com/> for redbook, residency, and workshop announcements.

Index

A

- AIX platform
 - allocate file system storage 240
 - create journal file system 240
 - create logical volume 239
 - creating groups 234
 - creating users 234
 - detailed product mapping 48
 - increasing free space 231
 - mount file system for CD ROM 235
 - runtime topology A product mapping 42
 - runtime topology B product mapping 46
 - unmount the CD-ROM (umount) 237
 - working example runtime environment 229
- Animated GIF Designer 175
- Application and architecture domains 155
- Application and data nodes 25
- Application client 77
- Application design
 - Application elements 106
 - Application structure 111
 - Controller 119
 - General Web application design 105
 - Page construction 119
 - Response time 122
 - Security 123
 - User tracking and session management 120
 - Where to find more information 151
 - Working logic 112
- Application Framework for e-business 7, 168
 - Key Architecture Elements 8
- Application server node 25
- Application topology
 - choosing 13
 - Enterprise-out 13
 - Web-up 13
- Application topology 1 14
 - business driver 14
 - considerations 15
 - example 15
 - key features 15
 - Web-up 13
- Application topology 2 15
 - business driver 16
 - considerations 18
 - Enterprise-out 13

- example 19
- key features 17
- Authentication 123, 213
 - Basic 123
 - X.509 123

B

- Backup and recovery guidelines 215
- Backup guidelines
 - application backup and recovery 220
 - backup and recovery 222
 - LDAP database backup 221
 - operating system 220
 - Tivoli Storage Manager 216
 - WebSphere database 221
- beans
 - See JavaBeans
- Blaze
 - Advisor Builder 174, 379, 380
 - considerations 380
 - Advisor Server 379
- Build cycle 159
- Business elements 161
- Business logic 114, 185

C

- C++
 - CGI 95
 - CGI development 272
 - compiler 177
 - modules 112
- category bean 335
- Category display 329
- child category list bean 335
- Client-side scripts 82
- Commands 91, 115, 183
- Commands, tasks, overridable functions 183
- Commerce application structure 112
- Commerce server node 25
- Configuration
 - AIX platform
 - WebSphere Commerce Suite 262
- Configure
 - Windows NT platform
 - DB2 client 433
- Configuring

- Windows NT platform
 - WebSphere Commerce Suite 443
- connectors 101
- Controller elements 161
- cookies 120
- Creating a database query 364
- Custom servlet and bean example 362
- Customizing a store 282

D

- Database server node 23
- DB2
 - AIX platform
 - installing DB2 client 251
 - installing DB2 server 230
 - start DB2 server 263
 - Control Center 182
 - Windows NT platform
 - installing DB2 client 432
 - installing DB2 server 423
- db2www.ini
 - include_path 282
 - macro_path 282
- Deployment 163
 - process 164
 - tools 164
 - work products 163
- Develop source code
 - process 161
 - tools 162
 - work products 160
- Development environment 168
- Development process
 - build cycle 159
 - deployment 163
 - macro design 156
 - micro design 158
 - overview 153
 - solution outline 155
- Development tools 170
 - Page Designer 173
 - VisualAge for Java 175
 - WebSphere Commerce Studio 171
- Directory and security services node 23
- Dispatcher node 24
- Domain Name Service (DNS) node 23
- Dynamic data sources beans 336
- Dynamic HTML (DHTML) 80

E

- e-Commerce application functionality 166
- Enterprise JavaBeans (EJB)
 - entity bean 88
 - session bean 88
 - technology options 87
- Enterprise-out 13
- Extensible Markup Language (XML) 80

F

- fixpak
 - DB2 and SecureWay Directory 449

H

- HotMedia 177
- HTML 79
- HTTP Server
 - Administration Server 182
- HTTP Server (Apache)
 - AIX platform
 - configure HTTP server name 250
 - create HTTP admin 246
 - create HTTP runtime user 247
 - enabling SSL 248
 - installing 244
 - start HTTP Administration Server 250
 - start HTTP Server 250
 - verify install and config 251
 - create HTTP admin
 - AIX platform 244
 - Windows NT platform
 - configure HTTP server name 431
 - create HTTP admin 429
 - enabling SSL 429
 - installing 427
- httpd.conf 281

I

- Installing
 - AIX platform
 - DB2 client installation 251
 - DB2 server installation 230
 - HTTP Server 244
 - JDK 1.1.8 241
 - MQSeries client 408
 - MQSeries server 403
 - Net.Data 259

- SecureWay Directory (LDAP) 294
- WebSphere Application Server 252
- WebSphere Commerce Suite 260
- Netscape Communicator 4.61 263
- Windows NT platform
 - DB2 client installation 432
 - DB2 server installation 423
 - HTTP Server (Apache) 427
 - JDK 1.1.7 426
 - Net.Data 440
 - SecureWay Directory (LDAP) 449
 - WebSphere Application Server 435
 - WebSphere Commerce Suite 441
- ITSO sample store 227
 - defined 227

J

- Java applets 82
- Java Message Service (JMS) 104
- Java Naming and Directory Interface (JNDI) 103
- Java Servlets 332
 - example 362
 - technology options 86
- Java technologies in Commerce Suite
 - overview 327
- Java Transaction API (JTA) 104
- JavaBeans 116, 334
 - example 362
 - technology options 87
- JavaScript 82
- JavaServer Page (JSP) 327
 - creating category list page 337
 - creating product list page 344
 - creating product page 347
 - deployment 357
 - development recommendations 361
 - example 337
 - execution monitor 176
 - technology options 86
 - testing 356
- JDBC 101
- JDK
 - AIX platform
 - installing 241
 - verify install 244
 - JDK 1.1.8 PTF 6 for AIX 242
 - selecting the version 241
 - Windows NT platform

- installing 426
- JScript 82

K

- Key Management 249
 - ikeman 249
 - start 249

L

- LDIF 304

M

- Macro design 156
- Markup languages 79
 - Dynamic HTML (DHTML) 80
 - HTML 79
 - XML, XSL 80
- merchant bean 335
- Micro design 158
- Model-View-Controller 111
 - controller 111
 - model 111
 - view 111
- MQSeries
 - AIX platform
 - installing MQSeries client 408
 - installing MQSeries server 403
 - back-end system simulation 418
 - example message flow 414
 - inbound XML messages 402
 - Create_NC_Customer 402
 - Update_NC_Customer 402
 - Update_NC_OrderStatus 402
 - Update_NC_ProductInventory 402
 - Update_NC_ProductPrice 402
 - integration with Commerce Suite 401
 - outbound message verification 415
 - outbound XML messages 402
 - Report_NC_PurchaseOrder 402
 - XML message example 413
- MQSeries adapter 410
 - enable MQSeries XML messages 412
 - enabling 411
 - outbound XML messages 412

N

- ncmqbridge 411

- Net.Data 95, 112
 - AIX platform
 - installing 259
 - Windows NT platform
 - installing 440
 - netcpswd file 262
 - Netscape Communicator
 - installing 263
 - Network Dispatcher 230
 - Node types 22
 - directory and security services node 23
 - Dispatcher node 24
 - DNS node 23
 - Public Key Infrastructure (PKI) node 23
 - User node 23
 - Web application server node 22
- O**
- Organization role 165
 - Business logic developer 165
 - Script developer 165
 - Third-tier integration developer 166
 - View developer 165
 - Overridable functions 92, 115, 184
- P**
- Page construction
 - Dynamic content 120
 - Static content 119
 - Templates + dynamic data 120
 - Page Designer 175
 - Patterns
 - benefits of Patterns for e-business 4
 - how to use the patterns 6
 - introduction 3
 - Patterns for e-business 3
 - User-to-Online Buying Pattern 5
 - Patterns for e-business 3
 - Application Integration Pattern 4
 - Business-to-Business Pattern 4
 - e-business solutions 11
 - User-to-Business Pattern 4
 - User-to-Data Pattern 4
 - User-to-Online Buying Pattern 4
 - User-to-User 4
 - Web site 6
 - Perfect Photo 177
 - Performance guidelines 53
 - Application development 72
 - Architecture and capacity planning 53
 - General 76
 - Operating system and network 54
 - Web application servers and technologies 56
 - Where to find more information 76
- Personalization 97**
- affiliate program 379
 - compile the rule project 389
 - configuration with Commerce Suite 393
 - create the rule project and ruleflow 384
 - create the ruleset and assign task 386
 - debug tips 399
 - example 382
 - inventory 379
 - legal policy 379
 - merchandising 379
 - overview 379
 - publish the project 392
 - time based 379
 - worksheets 383
- Presentation elements 160**
- Presentation logic 116, 185**
- comparison of Net.Data and JSPs 118
 - JavaServer Pages (JSP) 117
 - Net.Data macros 117
- price bean 335
 - Product Advisor 179
 - product attribute bean 335
 - product attribute list bean 335
 - product bean 335
 - product list bean 335
 - Product list display 329
 - Product mapping 39
 - runtime topology A 40
 - runtime topology B 43
 - Protocol firewall and domain firewall nodes 24
 - Public Key Infrastructure (PKI) node 23
- R**
- Remote Method Invocation (RMI) 103
 - Rules 97
 - Runtime environment
 - AIX platform 229
 - Windows NT platform 423
 - Runtime topology
 - choosing 21
 - emerging runtime topology 21

- proven runtime topology 21
- variations 21
- Runtime topology A 25
 - example 29
 - product mapping 40
 - AIX platform 42
 - Windows NT platform 40
- Runtime topology B 32
 - example 35
 - product mapping 43
 - AIX platform 46
 - Windows NT platform 43

S

- SecureWay Directory
 - add a suffix 302
 - Add entries to the directory database 304
 - AIX platform
 - installing 294
 - delete a suffix 303
- SecureWay Directory (LDAP)
 - add and delete suffixes 302
 - AIX platform
 - configuration 296
 - overview 289
 - configure with Commerce Suite 310
 - install verification 295
 - integration with Commerce Suite 308
 - LDIF 304
 - Management Tool (DMT) 317
 - Server Administration 301
 - Server Administration GUI 304
 - starting and stopping the server 301
 - starting the configuration utility 296
 - trouble shooting 326
 - what is LDAP? 290
 - why should I use LDAP? 290
 - Windows NT Platform
 - installing 449
- SecureWay Directory Management 203
- SecureWay Firewall 230
- Security guidelines 207
 - network security 209
 - operating systems security 208
 - physical systems security 207
 - Web application security 210
- servlets
 - See Java Servlets

- Session
 - clustering 121
 - management 120
 - persistence 121
- Shared file system node 24
- Site Analyzer 197
 - configuration 197
 - reporting 198
 - starting 197
- Solution outline 155
- Store Creator 172
 - create a store 273
- Store Profile Editor 173
- Systems management guidelines 187

T

- Tasks 92, 115, 184
- Technical walkthrough
 - comparison with use case 130
 - overview 129
- Technology options 77
 - Connectors 101
 - Java Servlets 99
 - JavaBeans 100
 - JavaServer Pages 98
 - JDBC and SQLJ 101
 - Mass import XML 102
 - MQSeries Adapter 102
 - Web clients 77
 - WebSphere Application Server 84
 - WebSphere Commerce Suite 89
 - Where to find more information 104
- Third-tier elements 161

U

- URL rewriting 120
- Use case
 - comparison with technical walkthrough 130
 - Example 130
- User node 23
- User Registry 124
- User tracking 120
- User-to-Online Buying Pattern 5
 - defined 5
 - examples 5
 - redbook structure 11

V

VisualAge for Java 175
configuration 272

W

Web application server node 22
Web browser 78
Web browser client 77
Web clients 77, 108
 Benefits of browser clients 108
Web server redirector node 24
WebArt Designer 175
WebSphere Application Server
 Administrative Console
 analyzing usage statistics 191
 configuring applications 190
 controlling access to applications 190
 daily administrative operations 190
 optimizing performance 191
 troubleshooting 191
 administrative domain 192
 administrative repository 192
 administrative resources 192
 administrative server 191
 Administrator's Console 182, 190, 192
 AIX platform
 configure default_host 258
 DB2 server configuration 253
 installing 252
 modify admin.config 257
 start Administrator's Console 258
 view tracefile log 257
 centralized administration 196
 configuration for Commerce Suite 336
 containment hierarchy 192
 node 191
 resources administration 194
 Site Analyzer 197
 topology 192
 verify installation 258
 Windows NT platform
 installing 435
 start server 439
 verify server is running 439
WebSphere Application Server Administrative Console
 analyzing performance 191
WebSphere Catalog Architect 181
WebSphere Commerce Studio 171, 269
 add a server 276
 create target file system
 AIX platform 277
 creating a bean and servlet 366
 creating a database query 364
 define target publishing host 275
 ftp publish method 278
 installation 271
 modify publishing paths 279
 modify storesDocRoot 278
 modify the store 276
 overview 270
 publish base store 280
 publishing files 370
 publishing projects 275
 set default server 276
WebSphere Commerce Suite
 add alias to httpd.conf file 281
 Administrator 178
 Site Manager 178
 Store Manager 179
 AIX platform
 configuration 262
 installing 260
 beans
 category bean 335
 child category list bean 335
 dynamic data source beans 335
 merchant bean 335
 price bean 335
 product attribute bean 335
 product bean 335
 product list attribute bean 335
 product list bean 335
 C++ CGI 95
 CGI implementation 94
 Commands 91
 Configuration Manager 178, 200
 Configuration Manager (start_config) 264
 create instance 264
 Database 94
 DB2 client configuration
 AIX platform 253
 deploying servlets 370
 design
 beans 113
 JSP 113
 servlets 113

- modify db2www.ini include and macro path 282
- Overridable functions 92
- personalization configuration 393
- SecureWay Directory configuration 310
- starting the administrator 443, 447
- Tasks 92
- verify install and config
 - AIX platform 268
 - Windows NT platform
 - configuring 443
 - installing 441
- WebSphere resource management 189
- WebSphere security model and policy 213
- WebSphere Test Environment 176
- Web-up 13
- Window NT platform
 - working example runtime environment 423
- Windows NT platform
 - detail product mapping 45
 - runtime topology B product mapping 43
- Working example
 - Commerce server node 229
 - Database server node 230
 - Directory node 230
 - Dispatcher node 230
 - Firewall nodes 230
 - Integration node 230

X

- XML 80, 85
 - MQSeries XML messages 402
- XSL 81, 85

IBM Redbooks review

Your feedback is valued by the Redbook authors. In particular we are interested in situations where a Redbook "made the difference" in a task or problem you encountered. Using one of the following methods, **please review the Redbook, addressing value, subject matter, structure, depth and quality as appropriate.**

- Use the online **Contact us** review redbook form found at ibm.com/redbooks
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redbook@us.ibm.com

Document Number	SG24-6156-00
Redbook Title	e-Commerce Patterns Using WebSphere Commerce Suite, Patterns for e-business Series
Review	
What other subjects would you like to see IBM Redbooks address?	
Please rate your overall satisfaction:	<input type="radio"/> Very Good <input type="radio"/> Good <input type="radio"/> Average <input type="radio"/> Poor
Please identify yourself as belonging to one of the following groups:	<input type="radio"/> Customer <input type="radio"/> Business Partner <input type="radio"/> Solution Developer <input type="radio"/> IBM, Lotus or Tivoli Employee <input type="radio"/> None of the above
Your email address: The data you provide here may be used to provide you with information from IBM or our business partners about our products, services or activities.	<input type="checkbox"/> Please do not use the information collected here for future marketing or promotional contacts or other communications beyond the scope of this transaction.
Questions about IBM's privacy policy?	The following link explains how we protect your personal information. ibm.com/privacy/yourprivacy/



Redbooks

**e-Commerce Patterns Using WebSphere
Commerce Suite, Patterns for e-business Series**

(1.0" spine)
0.875" <-> 1.5"
460 <-> 788 pages



e-Commerce Patterns Using WebSphere Commerce Suite

Patterns for e-business Series



Selecting the topology and products for your e-commerce solution

Patterns for e-business are a group of proven, reusable assets that can be used to increase the speed of developing and deploying Web applications. The pattern discussed in this book, User-to-Online Buying, identifies the interaction of users with enterprise Web sites that sell goods and services.

Guidelines for performance, design, and development

Part 1 of the redbook guides you through the process of selecting an application and runtime topology. The pattern provides you with platform-specific product mappings for implementing the chosen runtime topology.

Creating e-commerce sites by example using Java

Part 2 of the redbook provides a set of guidelines for building your e-commerce application. These guidelines include performance, technology options, application design, application development, systems management and security.

Part 3 of the redbook teaches you by example how to implement an e-commerce application using application topology 2. Some of the technology highlights include JSPs, servlets, beans, MQSeries XML messages, personalization, and LDAP.

**INTERNATIONAL
TECHNICAL
SUPPORT
ORGANIZATION**

**BUILDING TECHNICAL
INFORMATION BASED ON
PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:
ibm.com/redbooks**

SG24-6156-00

ISBN 0738417025