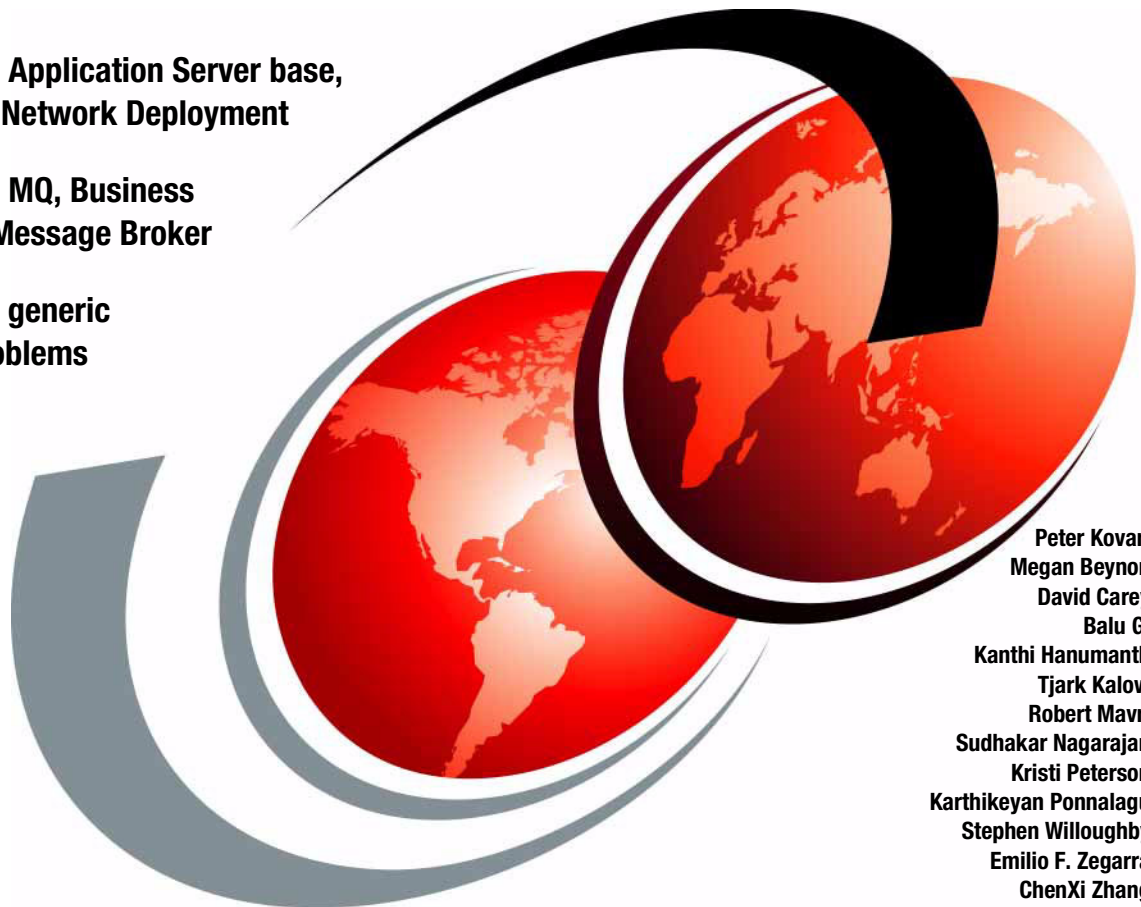


Problem Determination Across Multiple WebSphere Products AIX Platform

WebSphere Application Server base,
Enterprise, Network Deployment

WebSphere MQ, Business
Integrator Message Broker

Addressing generic
runtime problems



Peter Kovari
Megan Beynon
David Carey
Balu G.
Kanthi Hanumanth
Tjark Kalow
Robert Mavri
Sudhakar Nagarajan
Kristi Peterson
Karthikeyan Ponnalagu
Stephen Willoughby
Emilio F. Zegarra
ChenXi Zhang



International Technical Support Organization

**Problem Determination Across Multiple WebSphere
Products
AIX Platform**

April 2004

Note: Before using this information and the product it supports, read the information in “Notices” on page ix.

First Edition (April 2004)

This edition applies to AIX 5.2 Maintenance Level 2, WebSphere Application Server Enterprise V5.0.2, WebSphere MQ 5.3.0.4, WebSphere Business Integration Message Broker V5, WebSphere Studio Application Developer Integration Edition V5.0, WebSphere Studio Application Developer V5.1.

© Copyright International Business Machines Corporation 2004. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Notices	ix
Trademarks	x
Preface	xi
The team that wrote this redbook	xii
Become a published author	xvi
Comments welcome	xvi
Part 1. Introduction	1
Chapter 1. Introduction	3
1.1 What this book is about	4
1.2 Who this book is for	6
1.3 What about other problem determination books ?	7
1.4 How to use this book	7
1.5 Platform and products	8
Chapter 2. Guidelines for problem determination	9
2.1 Things to check	10
2.1.1 Responsibilities and problem ownership	10
2.1.2 Availability of resources	11
2.1.3 Processes	14
2.2 Problem determination steps	16
2.2.1 Identifying the problem	17
2.2.2 Documenting the problem	18
2.2.3 Organizing problem resolution	18
2.2.4 Analyzing the problem	19
2.2.5 Implementing the problem resolution	19
2.2.6 Closing the problem	20
2.3 Guidelines for customers	21
2.3.1 Defining the problem	21
2.3.2 Gathering background information	21
2.3.3 Gathering relevant diagnostic information	22
2.3.4 Determining problem severity	22
2.3.5 Gathering additional information	23
2.3.6 Reporting a software problem	23
2.3.7 Accessing Software Support	24
Chapter 3. The eMerge business scenario	27

3.1	The eMerge business scenario	29
3.2	Architecture	30
3.3	Application details	32
3.3.1	Get Customer Details use case	32
3.3.2	Request and Accept Quote use case	33
3.4	The Patterns for e-business approach	35
3.5	Products	38
3.5.1	Web server	38
3.5.2	WebSphere Application Server	38
3.5.3	Message Oriented Middleware	39
3.5.4	Integration middleware	39
3.5.5	Back-end applications	39
Part 2. Problem determination details		41
Chapter 4. Problem determination tools: runtime environment		43
4.1	Chapter organization	44
4.2	Operating system	44
4.3	WebSphere Application Server	52
4.3.1	JVM logs	52
4.3.2	Viewing the service log	56
4.3.3	Log Analyzer	56
4.3.4	Version information	57
4.3.5	Collector tool	58
4.3.6	Debugging with the Application Server Toolkit	59
4.3.7	WebSphere Application Server tracing	59
4.3.8	First Failure Data Capture tool	64
4.3.9	Other WebSphere Application Server logs	64
4.3.10	BackupConfig and RestoreConfig	65
4.3.11	DumpNameSpace	65
4.3.12	Tivoli® Performance Viewer	65
4.4	WebSphere MQ	66
4.4.1	First-Failure Support Technology (FFST™)	67
4.4.2	Error logs	68
4.4.3	WebSphere MQ error logs	69
4.4.4	Tracing WebSphere MQ for AIX	71
4.4.5	WebSphere MQ Java tracing	73
4.4.6	Formatting the MQ trace file	73
4.4.7	WebSphere MQ Explorer	75
4.5	WebSphere Business Integration Message Broker	80
4.5.1	Tracing	80
4.5.2	Message Broker Debugger	90
4.6	IBM HTTP Server	94

4.7	IBM DB2 UDB	94
4.7.1	Java Database Connector tracing	96
4.7.2	Running a JDBC trace	96
4.8	IBM Tivoli Directory Server	97
Chapter 5. Problem determination tools: development environment		101
5.1	WebSphere Studio	102
5.2	WebSphere Studio Application Developer Integration Edition V5.0.2	102
5.2.1	Application assembly	102
5.2.2	Compile	103
5.2.3	Component test perspective	103
5.2.4	IBM Agent Controller	103
5.2.5	Test environment	103
5.2.6	Debugger	106
5.2.7	Process debugger	107
5.2.8	Profiling perspective	110
5.3	WebSphere Studio Application Developer V5.1	113
5.3.1	Log and Trace Analyzer for Autonomic Computing	113
5.3.2	J9 JVM: Hot Method replace	116
5.3.3	Active Script debugging	117
5.3.4	SQLJ debugging	117
5.3.5	J2EE Code Validation technology preview tool	118
5.4	RFHUTIL	119
Chapter 6. WebSphere Application Server : plug-in		123
6.1	Web server plug-in	124
Chapter 7. WebSphere Application Server: base		135
7.1	WebSphere Application Server	136
7.1.1	General problem determination guidelines	136
7.1.2	Application servers	137
7.1.3	Administrative Console	141
7.1.4	wsadmin problems	143
7.1.5	Applications	144
7.1.6	Tracing	149
7.1.7	Clients	150
7.1.8	Resource providers	152
7.1.9	Resource providers: JDBC providers	152
7.1.10	Resource providers: data sources	155
7.1.11	Resource providers: WebSphere MQ JMS Provider resources	158
7.1.12	Embedded Messaging	159
7.1.13	Security	162
7.1.14	Security: SSL	165
7.1.15	Security: Java 2 security	167

7.1.16	Console users and groups	168
7.1.17	Naming	168
7.1.18	Java Virtual Machine: Memory	169
7.1.19	HTTP Sessions	171
7.1.20	Performance	171
Chapter 8. WebSphere Application Server Enterprise		173
8.1	WebSphere Application Server Enterprise	174
8.1.1	Process Choreographer	174
8.1.2	Business rule beans	190
8.1.3	Extended messaging	192
8.1.4	Asynchronous beans	197
8.1.5	Dynamic query	198
8.1.6	Scheduler	200
8.1.7	Startup beans	202
8.1.8	Application profiling	202
8.1.9	Object pools	203
8.1.10	WorkArea service	204
8.1.11	Internationalization service	205
8.1.12	Last participant support	205
8.1.13	ActivitySession service	207
8.1.14	Back-up cluster support	207
8.1.15	Administrative Console (Enterprise)	209
Chapter 9. WebSphere Application Server Network Deployment		211
9.1	WebSphere Application Server Network Deployment environment	212
9.1.1	Deployment Manager	212
9.1.2	Problems in the cell environment	215
9.1.3	Administrative Console	223
9.1.4	Workload management	225
9.1.5	HTTP session management	236
9.1.6	Resources	249
9.1.7	JMS Provider	250
9.1.8	Naming	251
9.1.9	ORB	255
Chapter 10. WebSphere MQ		263
10.1	WebSphere MQ	264
10.1.1	WebSphere MQ Messaging issues with the WebSphere Application Server	265
10.1.2	Client	271
10.1.3	Cluster	272
10.1.4	Channels	276
10.1.5	Security	278

Chapter 11. WebSphere Business Integration Message Broker	281
11.1 WebSphere Business Integration Message Broker	282
Chapter 12. The back-end diagnostic overview	301
12.1 WebSphere TXSeries CICS	302
12.1.1 The SYMREC file	302
12.1.2 Encina trace messages	303
12.1.3 DCE diagnostic messages	303
12.1.4 WebSphere TXSeries tracing	304
12.2 Encina tracing for CICS Application Server processes	305
12.2.1 Writing trace data to in-storage buffers	307
12.3 CICS Transaction Gateway and CICS Universal Client	307
12.4 CICS Universal Client tracing	309
12.4.1 Starting and stopping Client daemon tracing	309
12.4.2 Wrapping the Client daemon trace	311
12.5 Formatting the binary trace file (CICSFTRC)	311
12.6 Client daemon trace analysis	312
12.7 CICS Transaction Gateway tracing	313
12.8 CICS Transaction Gateway on z/OS	313
12.8.1 The Gateway daemon	314
12.8.2 Trace file allocation	314
12.8.3 Application trace	315
12.8.4 Gateway daemon trace	316
12.8.5 JNI tracing	317
12.8.6 EXCI trace	318
12.9 CICS problem diagnosis	319
12.9.1 CICS messages	319
12.9.2 CICS internal trace	320
12.9.3 CICS Trace Control Facility (CETR)	321
Part 3. Appendixes	323
Appendix A. Problem determination tools: other platforms	325
Windows 2000 Server	326
WebSphere MQ: Windows	328
WebSphere Business Integration Message Broker: Windows	330
z/OS	332
System Management User Interface debug utilities	332
Language Environment® (CEEDUMP)	334
WebSphere MQ: z/OS	334
WebSphere MQ channel trace	338
IPCS and WebSphere MQ	339
WebSphere Application Server: z/OS	340
Diagnostic data	341

Executing the CTRACE for WebSphere	343
WebSphere Business Integrator: z/OS	347
Components of WebSphere Business Integrator on z/OS	347
Capturing a dump of the relevant address spaces	349
Displaying the status of a trace	350
Collecting a user execution group trace	350
Collecting a service execution group trace	351
Logs	352
Useful HFS files	352
Trace files	353
LDAP: z/OS	353
IBM HTTP Server	355
General configuration	355
SSL handshake and configuration issues	356
CMS key database (.kdb) and certificate issues	357
LDAP authentication related issues	358
Server hang issues	360
Request failure (500, 404, 400, etc.) issues	362
Dynamic content, tracing IBM HTTP Server and WebSphere connection	363
Appendix B. Methodology	365
The methodology used in this book	366
Why do we need a methodology?	366
How does the methodology work?	366
Work products	367
Related publications	371
IBM Redbooks	371
Online resources	371
How to get IBM Redbooks	374
Help from IBM	374
Index	375

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described 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:
IBM Director of Licensing, IBM Corporation, North Castle Drive Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law. INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.


This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include 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.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

@server®	Domino®	OS/2®
@server®	DB2 Connect™	OS/390®
Redbooks (logo)  ™	DB2 Universal Database™	Passport Advantage®
alphaWorks®	DB2®	Redbooks™
developerWorks®	DRDA®	RACF®
ibm.com®	Encina®	S/390®
iSeries™	FFST™	SecureWay®
z/OS®	Informix®	SupportPac™
zSeries®	IBM®	SLC™
AIX®	IMS™	SP1®
Cloudscape™	Language Environment®	Tivoli®
CICS/ESA®	MQSeries®	TXSeries®
CICS®	MVS™	WebSphere®
CICSplex®	MVS/ESA™	XDE™

The following terms are trademarks of other companies:

Intel, Intel Inside (logos), MMX, and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Preface

This IBM Redbook provides detailed information about problem determination for various WebSphere® products in a real-life scenario. It is a valuable resource of information for IT administrators, IT specialists and application developers.

Part 1, “Introduction” on page 1 includes an explanation of how the book is organized and how it should be used for problem determination. It also contains details about the solution chosen for the book to better understand the environment and to provide a real-life scenario.

Part 2, “Problem determination details” on page 41 is the core of the whole book. This part starts with two chapters discussing problem determination tools, one chapter for the runtime environment (AIX®) and one for the development environment (mainly WebSphere Studio Application Developer V5). The following chapters go into details about problems (symptoms) with each product: WebSphere Application Server Enterprise V5, WebSphere MQ V5.3, and WebSphere Business Integration Message Broker V5.

The “Appendixes” on page 323 extend the book with problem determination tools on other platforms, mainly z/OS®. It also offers insight into the methodology used for this book to document problems, tools and problem determination techniques.



The team that wrote this redbook (left to right): Kanthi Hanumanth, Robert Mavri, Sudhakar Nagarajan, David Carey, Kristi Peterson, Tjark Kalow, ChenXi Zhang, Stephen Willoughby, Peter Kovari

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.

Peter Kovari is a WebSphere Specialist at the International Technical Support Organization, Raleigh Center. He writes extensively about all areas of WebSphere. His areas of expertise include e-business, e-commerce, security, various Internet technologies and mobile computing. Before joining the ITSO, he worked as an IT Specialist for IBM in Hungary.

Megan Beynon has been working at IBM for three years. She began her career in WebSphere MQSeries® Test and now works for Software Group System House. She is currently a Solution Tester specializing in WebSphere MQ and WebSphere Business Integration Message Broker low-level architecture, infrastructure and development. In this role, she works closely with architects, customers, and specialists in other areas. She also has the opportunity to interact closely with development, resolving defects and reviewing product design.

Kanthi Hanumanth is a Senior I/T Specialist with IBM Software Services for WebSphere, part of the IBM Software Group. Kanthi's diverse roles include providing consulting services, education and mentoring on J2EE technologies,

specifically WebSphere and WebSphere Studio products, to Fortune 500 clients. He holds a Master's degree in Computer Science from the Victoria University of Technology, Melbourne, Australia.

Robert Mavri is an Advisory IT Specialist working in IBM Global Services Slovenija as a member of the EMEA WebSphere Back Office team. Before joining IBM three years ago, he worked as a development manager with the team responsible for the development of CTI applications. Robert holds a Master's degree in Electrical Engineering from the University of Ljubljana. His computer career started with ZX81 about 20 years ago. His areas of expertise include Biomedical Signal Processing, Analysis and Modelling, Software development and object-oriented programming.

Sudhakar Nagarajan is an IBM Certified Specialist, presently team leader for the WebSphere Globalization testing team under the Software group at RTP. Prior to joining the Software group, he was an IT specialist under Global Services, working with various clients. His background includes over ten years of application design, development and project management on both mainframe-based and distributed systems across a wide variety of industries and platforms. He holds a Master's degree in Manufacturing Engineering from REC Tiruchy, India.

David Carey is a Senior IT Advisory Specialist with the IBM Support Center in Sydney, Australia, where he provides defect and non-defect support for CICS®, CICSplex/SM, the WebSphere MQ family of products, and z/OS. David has 24 years of experience within the information technology industry, and has written extensively about diagnostic processes for the ITSO.

Kristi Peterson is a software engineer from Rochester, MN. She has three years of experience in the field of software testing. Kristi holds degrees in Computer Science and English from Luther College in Decorah, Iowa. Her areas of expertise include WebSphere Application Server, software testing, documentation review, security and scenario testing development.

Tjark Kalow is an IT Architect with Application Management Services (AMS) in IBM® Global Services in Germany. He has six years of project experience in e-business technology and WebSphere solutions for customers. He holds a degree in Business Computing from the University of Paderborn, Germany. His areas of expertise include e-business technologies and solutions, WebSphere Application Server and Portal Server, Java™ 2 Platform Enterprise Edition, and the IBM Global Services Method.

ChenXi Zhang is an Advisory IT Specialist in IBM China. She has five years of experience in WebSphere technical support. She provides the official support for WebSphere problems originating in China, both defect and non-defect. Her

areas of expertise include WebSphere Application Server, WebSphere Portal Server and WebSphere Studio Application Developer.

Stephen Willoughby is a Solution Test specialist working for the Hursley Solution Test Center, part of the WebSphere Platform System House. He joined IBM in 2000 for 15 months as part of his studies before returning to IBM as a full-time employee following graduation from the University of York in 2002. Since his return, he has developed J2EE software using IBM's WebSphere Studio Application Developer and WebSphere Studio Application Developer Integration Edition tools, including parts of the eMerge scenario used in this book. Stephen regularly comes across many problems which must be solved when deploying and testing scenarios in his work as a solution tester. Stephen has also had software published on IBM's alphaWorks® site.

Emilio Zegarra is a Software Engineer in the IBM WebSphere Competency Center in Pittsburgh, PA. His areas of expertise include User Interface design (for which he holds a patent), and J2EE, Web Services, and object-oriented development. He is an IBM Certified Systems Expert and Administrator on WebSphere Application Server, and an IBM Certified Specialist on WebSphere Studio Application Developer. Emilio holds a B.S. in Industrial Engineering from Purdue University and an M.S. in Industrial Engineering from the University of Pittsburgh. He is currently pursuing an M.S. in Information Technology - Software Engineering from Carnegie Mellon University.

Karthikeyan Ponnalagu is a Software Engineer at the IBM Software lab in Bangalore, India. He has five years of experience in software development. He is currently involved in architecting integration solutions for WebSphere Commerce. He has co-authored a redbook on EJB Mapping Techniques. He also co-authored a paper, presented in the AOSD conference, conducted by the IBM Academy of Technology. He has participated in IBM developerWorks® conferences as a speaker to discuss Web Services. He holds a degree in Electronics and Communication Engineering. His areas of interest include object-oriented languages, distributed system architecture, and aspect-oriented programming.

Balu G. is a Staff Software Engineer with IBM JDK Support team in India Software Labs, IBM Global Services India Pvt Ltd, Bangalore, India. He has been working with the JVM support team for last three years and is currently supporting ORB component. Prior to joining IBM, he accumulated almost three years of experience in application programming using C++ in domains like CAD/CAM and digital manufacturing. He holds an MTech in Mechanical Engineering.

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

Carla Sadtler
Cecilia Bardy
Gail Christensen
Jeanne Tucker
Margaret Ticknor
Michele Galic

International Technical Support Organization, Raleigh Center

Andre Hoehener
Andrew Leonard
Charlene Sun
Chris D. Johnson
David Zavala
Denis Masic
Eric Labadie
Ewan Withers
Jon K. Peterson
Lee Perlov
Logan Colby
Mary Trumble
Megan Beynon
Paul Titheridge
Rajeshkumar N Danda
Srecko Janjic
Vernon Green

Special thanks to the **IBM Software Group Collaboration Center Technical Team** for their feedbacks and comments.

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our Redbooks™ to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- ▶ Use the online **Contact us** review redbook form found at:

ibm.com/redbooks

- ▶ Send your comments in an Internet note to:

redbook@us.ibm.com

- ▶ Mail your comments to:

IBM Corporation, International Technical Support Organization
Dept. HZ8 Building 662
P.O. Box 12195
Research Triangle Park, NC 27709-2195



Part 1

Introduction



Introduction

Problems are a daily part of the information technology business. Which organization could safely say that their systems or applications always run without any problems whatsoever? However, in the past few years, the proliferation of e-business technologies, fueled by the success of the Internet, has led to widespread adoption of the underlying computing model, creating distributed systems that need to be available 24x7 and in which problems are often immediately and directly visible to the customer, for example because an error message is prominently displayed on a Web site. Furthermore, today's commercial software products are more complex. As with any new and complex technologies, experience in using them cannot be supplanted by anything else. This becomes immediately apparent when problems occur which have to be analyzed and solved within short time frames. Therefore, problem determination is a process which is crucial to the successful operation of a modern business.

1.1 What this book is about

This book deals both with the art of problem determination and a collection of problem determination tools and techniques for various IBM WebSphere software products. With regard to the latter, we will also deal with specific tips or general approaches for problem solving. The IBM products we will cover in detail are (see 1.5, “Platform and products” on page 8 for details):

- ▶ WebSphere Application Server Enterprise
- ▶ WebSphere MQ
- ▶ WebSphere Business Integration Message Broker

Some other IBM products will be mentioned and used in examples throughout this book, but are not covered or only covered broadly in terms of problem determination. Among these products are:

- ▶ WebSphere Studio Application Developer Integration Edition
- ▶ WebSphere Studio Application Developer
- ▶ IBM HTTP Server
- ▶ IBM DB2® Universal Database™
- ▶ TXSeries®
- ▶ CICS Transaction Gateway
- ▶ CICS/TS

We will not cover configuration and installation problems here, but instead focus on problems that may occur after the system has been successfully running at least once. There might be a couple of exceptional instances where we decided to document some of the most common installation and configuration problems.

Since problem determination for the products listed is already a wide scope for one book, we will generally not write about problem prevention techniques. Problem prevention involves using different techniques with multiple products and would widen the book’s scope too much. However, we strongly advocate active problem prevention techniques, one of which is actually a well-defined and thoroughly followed problem determination process in which previous problems are documented and analyzed so that they can be prevented in the future.

This book will not deal with tuning your production environment. Problem determination and tuning are closely related topics, each having the same outcome: a better performing product. You might perceive tuning as a subset of problem determination. Understanding the difference between problem determination and tuning is important. Knowing when to use tuning and when to use problem determination will save you time.

Problem determination is the process of determining the source of a problem, for example, a program component, machine failure, telecommunication facilities, user or contractor-installed programs or equipment, environmental failure such as a power loss, or user error.

Tuning is the process of adjusting an application or a system to operate in a more efficient manner in the work environment of a particular installation.

In other words, problem determination fixes functional problems, while tuning alleviates problems associated with slow processes.

It is important to be aware of one of the basic fundamentals of efficient problem diagnosis, sometimes called the "process of elimination" methodology. This implies that you may require additional diagnostic data to be captured using more refined and granular techniques as the analysis progresses.

The diagnostic data collection process is complex and hopefully this document will assist with the process of collecting sufficient initial data to speed up the resolution process.

The structure of the book

- ▶ Part 1, "Introduction" on page 1 begins with this chapter and contains generic information about problem determination.

In Chapter 2, "Guidelines for problem determination" on page 9, we discuss generic guidelines for problem determination. These guidelines capture professional knowledge about preparing for and performing problem determination and are not targeted to one specific product.

The business scenario which we used for most of the examples throughout the book is outlined in Chapter 3, "The eMerge business scenario" on page 27. We also describe how to narrow down a problem to a specific product or component in this chapter.

- ▶ Part 2, "Problem determination details" on page 41 contains product-specific problem determination information.

In Appendix 4, "Problem determination tools: runtime environment" on page 43, we describe various tools for problem determination in the runtime environment. You will find that the runtime environment is mostly narrowed down to the AIX and WebSphere platforms. Most of these tools are included with the IBM products covered in this book or are downloadable from the IBM Web site.

Appendix 5, "Problem determination tools: development environment" on page 101 deals with problem determination in the development environment. The focus of this chapter is on using IBM WebSphere Studio Application Developer.

Problem determination for IBM WebSphere Application Server Enterprise is split into multiple chapters. Chapter 6, “WebSphere Application Server : plug-in” on page 123 deals with the WebSphere Application Server HTTP Server plug-in, Chapter 7, “WebSphere Application Server: base” on page 135 with the Base product, Chapter 8, “WebSphere Application Server Enterprise” on page 173 with the Enterprise components and Chapter 9, “WebSphere Application Server Network Deployment” on page 211 with the Network Deployment parts.

Chapter 10, “WebSphere MQ” on page 263 contains information about determining problems in WebSphere MQ.

We describe problem determination for WebSphere Business Integration Message Broker in Chapter 11, “WebSphere Business Integration Message Broker” on page 281.

We also cover problem determination for some typical back-end products in Chapter 12, “The back-end diagnostic overview” on page 301, although the back-end applications and systems are outside of this book’s scope.

- ▶ Part 3, “Appendixes” on page 323 includes the relevant appendixes.

While developing the redbook, the team came across tools that are also available on other platforms. Appendix A, “Problem determination tools: other platforms” on page 325 includes tools that are neither AIX nor WebSphere tools. This appendix has an extensive z/OS tools section.

Appendix B, “Methodology” on page 365 describes the strategy we used in this book to document the key problem determination areas.

1.2 Who this book is for

The intended audience for this book includes professionals in the information technology industry who already have some experience with the products covered in this book. Preferably, you should also not be completely new to the topic of problem determination. We will give instructions and sometimes examples in this book, but they will not be detailed enough to make sense to someone unfamiliar with the products. We do assume product knowledge and experience on your part. If you have little or no knowledge about the products, please contact IBM Learning Services for available courses or check for other IBM Redbooks to enhance your product-related skills.

For this intended audience, this book will be useful when setting up a generic problem determination strategy and for specific problem determination with the products covered. We hope that you will find this book to be a useful reference for your daily problem determination tasks, although we certainly wish you problem-free work!

1.3 What about other problem determination books ?

As you can imagine, diagnostics is a huge area of problem determination in IT. The difficulty with problem determination is that there is an infinite number of possibilities for problems to arise. We simply cannot identify all the problems and work backwards to find determination techniques and tools for them. We have to provide an extensive set of tools, strategies and methodologies to help problem determination. But using tools and techniques from books is just one step. Real problem determination techniques have to be developed individually. Problem determination skills are developed over a long time and with lots of practice; we recommend that you follow this path and try to build this skill instead of searching for the ultimate tool or strategy that will solve everything for you.

The purpose of this book is to provide you with a collection of problem determination tools, techniques and tricks. This book can only include a small percentage of all the possible tools and problems. Knowing these limitations, we have tried to come up with a reasonable scope for this redbook.

The tools and problems described and documented here can be found in other places and in other documents. What this book does is collect numerous tools and problems and arrange them in a well-organized structure, a scenario that the reader can recognize and follow easily. The value of this redbook lies in its ability to organize and structure the existing knowledge in this area into one source.

If you cannot find a particular tool, a particular problem or a solution to a particular problem, do not hesitate to look up the product documentation. All of the products in this book have proper, extensive problem determination or troubleshooting guides included in the product documentation. It is not this book's intention to duplicate the existing documentation or replace any of it, so do not neglect this valuable source.

1.4 How to use this book

This book is intended to be used as a reference guide. It is structured in such a way that problems (symptoms) will be easy to find. You can read more about the methodology (how the book is organized) in Appendix B, "Methodology" on page 365. The methodology is not about problem determination methodology, but about how to document problems in an e-business environment.

We know that it is impossible to cover every problem, or even every type of problem, and this is not the intention of this book. This redbook collects problem determination tools and techniques, also problems (symptoms) in a specific environment (AIX) for a specific solution and scenario (the eMerge scenario).

In order to use this book efficiently, the reader should familiarize himself/herself with the scenario used in it; details can be found in Chapter 3, “The eMerge business scenario” on page 27 which provides the details of the scenario. Then, the reader should look for the similarities and patterns between the actual scenario where he/she needs to perform problem determination and the scenario given in this book. Obviously, this book is best organized for those who are dealing with a company merger scenario similar to the eMerge scenario.

Of course, for other readers who are dealing with different solutions but using the same platforms (in this case AIX, WebSphere), this redbook is still a very good source of information.

1.5 Platform and products

This redbook focuses mainly on the IBM AIX platform, with the following details:

- ▶ AIX V5.2 with Maintenance Level 01 (5200-01)

The products used in the scenario are:

- ▶ WebSphere Application Server Enterprise V5.0 with Fix Pack 2
- ▶ WebSphere MQ 5.3 with CSD 4
- ▶ WebSphere Business Integrator Message Broker V5.0 with CSD 2 (the previous version of this product was known as WebSphere MQ Integrator Broker)

The following products are used for running the test environment in the development environment. These products also have problem determination tools included.

- ▶ WebSphere Studio Application Developer Integration Edition V5.0
- ▶ WebSphere Studio Application Developer V5.1



Guidelines for problem determination

In this chapter, we talk about best practices for problem determination, problem analysis and, to a certain extent, problem solving. These best practices are based on IBM's vast experience with IT projects. Hopefully, you already follow some or all of these practices. However, it is never wrong to check again that you are prepared for dealing with a problem. Therefore, we will begin the chapter with some basic things to check. Then we will describe a generic approach to problem determination. Finally, we will give you some guidelines for interacting with IBM customer support.

Excellent problem determination skills cannot be acquired simply by reading a book like this one. They mostly come from experience as well as discipline. After reading this chapter, you should have a general understanding of a structured problem determination approach, which you can then apply in your daily work to gain more experience. At the completion of this chapter, you should be able to:

- ▶ Prepare for problem determination
- ▶ Adopt a systematic and thorough approach to dealing with problems
- ▶ Identify the different types of problems
- ▶ Escalate the problem to the IBM Support Center if required

2.1 Things to check

Common experience shows that not all problems can be anticipated in a production environment, even if it is exceptionally well managed. Therefore, the first and foremost best practice is to be prepared for problems to occur. Being prepared can also be regarded as an important problem prevention technique, because it can focus your attention on possible future problem areas or causes.

Following these basic rules will ensure that you can react in a timely and well-coordinated fashion when a problem occurs. This will make your life easier, but problem determination will always be a challenging and sometimes stressful task, because there is simply no good time for a problem to occur. Not following the best practices means that you will have to make decisions about these best practices or implement them the moment the problem occurs. This will at worst be impossible, which poses a clear danger for your production environment; at best, it will slow down your problem determination process, which you want to avoid.

Some of these best practices may seem trivial to you, because you have already implemented them. However, our experience shows that some things are easily overlooked even by the most seasoned professionals as long as no problem occurs. Therefore, we have included even the simpler practices from our experience.

2.1.1 Responsibilities and problem ownership

Dealing with a problem situation without clearly defined responsibilities is a tedious and unproductive way of working. You should consider defining the role of *problem owner* in your organization. The person filling out the role can be a different one in each case and be assigned only when the problem occurs. If it makes sense for certain production environments, the role can also be permanently assigned. Whichever way you choose, make sure that the coverage of the role is sufficient; this means that there is a qualified problem owner available when the problem occurs.

The problem owner is responsible for coordinating all resources in problem determination. Usually, these resources will be technical experts. The problem owner can and often should be a technical expert himself, but must still have the time needed for the coordination tasks. Therefore, basic project management skills should be in that person's list of skills.

It is important to know who the problem owner is and who else is responsible for all other tasks to be performed when dealing with the problem. The problem owner will then assign tasks to the available resources and will be the central communication hub. He must make sure that problem determination is

undertaken with a sense of urgency, but without panic, and that all actions are clearly communicated to all persons with a need to know. If necessary, he must also shield the team from management or external pressure, which often happens when the problem is externally visible.

Without an assigned problem owner or clear communication, problem determination tasks might start trailing off in the wrong direction or remain undone entirely, or counterproductive actions might even be performed.

Knowing who is responsible for what is also important for following best practices.

2.1.2 Availability of resources

Problem determination will very often be performed in time-critical situations. Therefore, you should make sure that you have the right resources prepared and available when it counts.

People and skills

The most important resource in problem determination is people with the right skills. You should be aware of the skills the people in your organization have, preferably by using a system to document the skills. Carefully review your production environment to determine the most critical components in which problems could occur. Then list the skills you would need to deal with a problem in each component. Map these skills onto the available skills. If there are gaps or limited critical resources, set up a plan to fill these gaps. This could be done by hiring new employees, training the existing employees, or by using external support (see below).

Even when you have the right skills available, make sure they will have the time to deal with problems in addition to their normal daily tasks. We have seen production environments where all necessary skills were in place, but were not fully available for problem determination due to a very high normal daily workload. In such an environment, the problem owner may have a hard time freeing the critical resources for problem determination.

External resources and support

Some problem determination tasks might depend on support from external resources. An example would be problems in components for which you have no skills in your organization and for which you have determined that building up those skills in your organization would not be feasible.

Determine which and how many external resources you need to have available, if you need them to be always at your site or only come in if needed and how fast

or for how long you would need them in case of a problem. All these factors determine the kind of contract you need to make with the supplier of the external resources. Of course, this contract has costs attached to it, maybe even when you never actually call in the external resources. But weigh the cost of that contract against the cost of possible delays in problem determination when you cannot get the external resources in time due to limitations of a cheaper contract.

Another type of external support is product support. Different vendors have very different policies for product support. Also, product support is available on different quality levels, with the higher levels obviously costing more. You should determine the level of support needed for each product in your production environment, be it hardware or software. Then compare this to the current level of support provided by the vendor to you and decide whether you need additional support contracts.

Usually, a vendor will not provide product support for older products, or the support will not be free anymore. Therefore, you should know the end of service (EOD) dates of the products you are using. For IBM software products on the AIX platform, you can look up the EOD on the Web site:

<http://www-1.ibm.com/services/s1/swm/>

Some vendors might allow you to escalate a problem to a higher contractual level of support on an ad-hoc basis for a fee, even if you do not have a regular contract for that level of support. If you know from past experience that problems seldom occur in one of your components, but are severe when they occur, this option might be useful for you, if the particular vendor offers it. However, if problems occur more often, a regular contract for a higher support level might be the better choice.

Make sure the telephone and fax numbers or e-mail addresses for the vendor's support hotline are available to the employees dealing with the problem. This documentation should also include the available hours of support, if it is not available 24x7, and any known limitations to the level of support, if applicable. Interaction with the external support should always be communicated to the problem owner, who will decide when to call external support. Always make sure to give the vendor the problem owner's name or the name of the person dealing with the particular problem area and prepare for that person to be reachable should the vendor try to contact him or her. Avoid a situation, where the external vendor has to deal with different people in your organization who do not communicate with each other.

Technical resources

Problem determination might require recreating the problem situation in another environment. A production environment can naturally not be altered and worked with in the same manner as a test environment. Therefore, you should have at

least one test environment which is as near in size and configuration to the production environment as possible. This test environment can then be used to recreate the problem situation and to try to reproduce the problem. Possible problem resolutions can be tested. This becomes especially important when the solution will change or delete data. With only the production environment at hand, the solution cannot be tested before applying it, leading to potential data corruption or loss. All changes to the production environment must be reproduced in the test environment. Preferably, this should be done in parallel. This particular test environment can therefore not be a developer test environment, because it must stay unaltered in correlation to the production environment. Please see 2.1.3, “Processes” on page 14 for details.

Another reason to have such a test environment is that some problem determination tasks might put the system under heavy load, and you cannot afford that on a production system. For example, some traces can use up many CPU cycles and also lots of hard drive space.

Make sure the test environment is readily available at all times when you might need it. It is a critical resource because of its linkage to the production environment.

There might be additional technical resources needed for problem determination, which are not part of the test environment as such. Examples include installation media, administrator workstations, file-transfer functionality and telecommunications. It is impossible to give a complete list here, because the needed resources depend on the particular production environment.

Documentation

Product documentation comes in very different paper-based and electronic formats. The documentation can come in very handy in problem determination, so you want to have it available. Especially with electronic documentation, this is sometimes harder to accomplish than it sounds. For example, you may have installed the documentation on your workstation, but if you then need to determine a problem directly in the data center, you might not be able to access your workstation from there. In this case, old-fashioned printed documentation might be needed.

Be sure to have the complete product documentation available in a format and on a medium that is suitable for the possible problem determination tasks. In some cases where only electronic documentation is available, you might want to purchase one or more sets of additional printed documentation from the vendor or print the documentation from the electronic version.

You might want to familiarize yourself with the way the documentation is organized. For example, are error messages listed in each part of the

documentation or is there a separate volume for error messages? Look especially for those parts of the documentation which deal with problem determination.

As to the question of whether you should use the electronic version of the documentation on the production environment, there are differing opinions. Of course, it comes in handy for problem determination, but it could also help an intruder on the system to exploit it better. Whether this is an actual security risk in your environment can only be determined by you.

Another type of documentation that should be available is documentation about previous problem occurrences and the actions taken at the time (please see also 2.1.3, “Processes” on page 14).

2.1.3 Processes

Even if all necessary resources for problem determination are readily available, they must still be utilized in a useful way, which can best be achieved with well-defined business processes. An overarching business process for problem determination will include various sub-processes. These processes must be individually defined for every organization, so we can only give guidelines for some sub-processes here.

Documentation

Documentation is perhaps the most often forgotten best practice, and not only in problem determination. All information collected during problem determination should be documented. This includes not only information about things you have learned or found out about the problem, but also about the steps you undertook. Some examples are:

- ▶ Problem context
- ▶ Problem description
- ▶ Problem owner and team working on the problem
- ▶ Problem status
- ▶ Actions taken
- ▶ Solution (if found)
- ▶ Recommendations for configuration changes, etc.

Documenting problem determination in all stages from the occurrence of the problem to its solution has multiple advantages:

- ▶ The problem is known in the context in which it occurred. Similar problem situations can be better analyzed if there is previous knowledge about a certain type of problem or symptom.

- ▶ The people involved in problem determination are known. This information can be used by other team members during problem determination or in later problem situations.
- ▶ The problem's status is obvious at all times. Problems cannot be forgotten this way. Even if the problem just went away by itself and was never solved, this information will not be lost.
- ▶ Lessons learned from this problem can be utilized in daily work or future projects.
- ▶ The information gathered does exist outside of people's heads and can be made available at all times.

Depending on the size of your organization, different media for documenting a problem might be used, from simple documents to a sophisticated problem database or even an expert system.

Test configuration

As we have mentioned in 2.1.2, "Availability of resources" on page 11, it is recommended to have a test environment to recreate problem situations for further analysis. This environment cannot be open to anyone because of its critical role and must be carefully managed with an appropriate process. This process must ensure the following.

- ▶ Only authorized persons can change the configuration.
- ▶ The configuration is always synchronized with the production environment's configuration, meaning that all changes in the production environment are reproduced in the test environment in a timely manner.
- ▶ No changes are made that would break the synchronicity with the production environment, unless this is necessary to test a possible solution to a problem.
- ▶ All changes are documented and can be traced back to the requirement that triggered their implementation.
- ▶ The test environment is always in a state suitable for reproducing a problem situation.

Escalation

The problem owner is responsible for judging the severity of the problem situation. There should be a process the user can follow to determine when, against whom and, most important of all, how a problem should be escalated. Some possible addressees of an escalation include:

- ▶ Vendors of a component, parts of the application or the infrastructure or other services.
- ▶ Internal departments delivering parts of the application or the infrastructure.

- ▶ Management.
- ▶ Spokespeople for the company (for problems that have external visibility).
- ▶ Law enforcement agencies (if a criminal security breach is part of the problem).

It is not useful to treat every problem situation as a major crisis. For example, management will surely not approve if every small problem is escalated to them. However, depending on the problem severity and the problem's effects on external entities, an escalation might be necessary. For this situation, an escalation process is helpful, so that the right persons or departments are contacted with the right information.

Emergency situation

An emergency plan is more than a special case of escalating a problem. The nature of an emergency means that the normal rules set in the other processes might not apply in the same manner. Dealing with an emergency might include different, maybe even drastic steps. You want to have a process for this, so that you are prepared even for the worst case scenario. The process in case of an emergency should also include the necessary steps to determine if the situation is actually an emergency. Apart from helping to preventing a panic reaction, this is also a kind of self-protection for the people involved, who might otherwise be reluctant to declare an emergency in case they might be later reprimanded for causing a false alarm.

2.2 Problem determination steps

This section outlines problem diagnosis fundamentals, which are not specific to one product. Other chapters in this book cover product-specific analysis methodologies for WebSphere Application Server, WebSphere MQ and WebSphere Business Integration Message Broker on the IBM AIX platform.

The steps that are taken to investigate and analyze a problem are outlined as follows:

1. Identify the problem
2. Document the problem
3. Organize problem resolution
4. Analyze the problem
5. Implement the problem resolution
6. Close the problem

2.2.1 Identifying the problem

A system problem can be described as any problem on your system that causes work to be stopped or degraded. The steps involved in diagnosing these problems are different for each type of problem. Before you can begin to diagnose a system problem, however, you have to know what kind of problem you have. Problem identification is often not a straightforward process, but an investigative exercise that requires a structured method which will enable the correct initial assessment to be made. This initial phase is important because decisions you make now relating to diagnostic data collection will influence the speed of finding the resolution.

The most important questions you must ask include:

- ▶ How did you first notice the problem? Did you do anything different that made you notice the problem?
- ▶ Is the process that is causing the problem a new procedure, or has it worked successfully before?
- ▶ If it was an existing procedure that was previously successful, what has changed? This refers to any type of change made to the system, ranging from adding new hardware or software, to configuration changes to existing software.
- ▶ What was the first symptom of the problem you witnessed? Were there other symptoms occurring around that point in time?
- ▶ Does the same problem occur elsewhere? Is only one machine experiencing the problem or are multiple machines experiencing the same problem?
- ▶ What messages are being generated that could indicate what the problem is?
- ▶ Can the failure be reproduced, and if so, what steps are being performed?
- ▶ Has the failing process generated an output of any kind, for example a memory dump?

All of these questions will enable you to develop an appropriate plan of action to aid with the resolution. You can never be criticized for providing too much diagnostic data, but too little information only delays the solving or escalation of the problem.

Be aware that symptoms clearly indicating a specific problem might be caused by another, yet undetected problem. Problems can be hidden by other problems. When you fix the most visible problem, another one may come to light. The problems that are unearthed during the problem determination process may be related to the one that was initially reported, in other words, these may be multiple problems with the same symptoms. In some cases, you may discover problems that are completely unrelated to the one that was initially reported. Use

the information in the others chapters of this book to drill down to the most likely problem, based on the information gathered from the questions above and from the problem identification steps described in the other chapters.

2.2.2 Documenting the problem

As outlined in 2.1.3, “Processes” on page 14, documentation of the problem and the analysis steps taken can assist not only with initial resolution, but also if the problem occurs again. In this step, documentation means all information available at this point. In this way, the problem is officially recognized and the next steps can be taken.

Significant information uncovered during problem determination should always be appended to the documentation. For larger, more complex problems, regular documentation during the analysis process can highlight areas that will become more crucial as the investigation progresses. This will enable you to develop a flow chart and reference point listing that can be referred to throughout your analysis. Make sure to document the final resolution for future reference.

2.2.3 Organizing problem resolution

Once the problem has been identified and documented, a problem owner can be assigned. Alert all the necessary resources that you have taken ownership of the problem and that their cooperation is required. Then assign further tasks.

Your prime objective as a problem owner is to ensure system availability. In the event of a major subsystem failure, your focus will be on the speedy restoration of the system. Subsystem failures will often generate their own diagnostic data, and the recovery process is often fairly straightforward. These systems will generally perform cleanup processes during recovery and system availability will be resumed. If the subsystem fails during the recovery then immediate problem analysis and resolution will be required.

Some assistance may be required if you are making little progress with your diagnosis. What you and your management are seeking is a speedy resolution, and it is far more professional to use all the facilities and technical expertise available to assist you. When the problem is related to an IBM product, the IBM Support Center is there to assist you. The analysis steps you have already performed and the diagnostic data you have collected will be of great help to the Support Center when they review the problem.

2.2.4 Analyzing the problem

Before you start the more complex, product-specific analysis procedures, you should review all of the data you currently have that may solve your problem. For example, make sure that you have:

1. Looked in the log files for any relevant messages or abnormal process termination information.
2. Reviewed the meaning of any error messages or codes in the relevant manuals.
3. Reviewed the system error log which contains information about hardware and software failures.

Problem analysis is a skill that develops the more you apply it. Unfortunately, problems vary in their complexity and frequency, and it is possible that tasks requiring this type of review may be infrequent in your environment. Of course, the ultimate aim is to have little need to perform complex problem diagnosis. This is why a sound methodology is necessary to assist with your analysis.

It is necessary to retain a focus during the analysis process and be aware that there are often alternative ways to approach a problem. To ask for assistance with a problem is not a sign of failure, but an indication you are aware that another person's views could speed up the resolution. A fresh idea can often stimulate and enhance your current thought processes.

Solving a problem is a combination of:

1. Your ability to understand the problem.
2. The quality of the diagnostic data you can review.
3. Your ability to use the diagnostic tools at your disposal.

2.2.5 Implementing the problem resolution

Successful diagnosis of the problem will result in a number of possible resolution strategies, depending on the type of problem:

- ▶ *User error* - This will require the user to correct their procedure to ensure a satisfactory resolution is implemented. If their procedure is impacting other users, then it is imperative that their prompt action be encouraged.
- ▶ *Software installation or configuration error* - You must ensure that all installation procedures have been correctly executed and any required customization has been performed correctly. You can use the test environment to verify the necessary steps. Until you can be sure of a successful implementation, it is advisable to remove this software, or regress to a previous level of the software until more extensive testing can be done in an environment that will not impact production workloads.

- ▶ *Software product fault* - If the fault is identified as a failure in software, a fix might already have been developed to solve this problem. In IBM terms, this fix is identified either as an Interim Fix or a Program Temporary Fix (PTF) and will need to be installed onto your system. Be aware that Interim Fixes have not been integration-tested and should only be used when IBM Support advises you to do so. Usually, you will want to test the Interim Fix or PTF in the test environment before applying it to the production environment. If the problem is causing a major impact, it is suggested that you expedite your normal fix application process and promote the fix to the problem system to hopefully stabilize that environment.

If the problem has not been previously reported to IBM, but is considered to be a software product fault, an authorized program analysis report (APAR) will be created and a solution will be provided in terms of a workaround, an Interim Fix or a PTF. The problem resolution is therefore not complete until the Interim Fix or PTF has been delivered and installed. In some cases, a workaround might be needed in the meantime.

- ▶ *Hardware fault* - If this is the problem, the resolution will be controlled by the hardware service representative, but may require some reconfiguration tasks on your part, depending on the nature of the problem. Consultation with the hardware vendor's service representative will clarify the requirements.
- ▶ *External fault* - A problem that is caused by external influences which are not external user errors is rare. Many unlikely, but possibly catastrophic faults fall into this category, for example power outages and bad weather conditions. The resolution will depend on the external entity that is causing the problem.

2.2.6 Closing the problem

When you have tested and implemented the problem resolution, ensure that all parties involved with this problem are informed of the closure of this issue. Finally, update the problem documentation. If the problem resolution included changes to configurations or processes, make sure the related operational procedures are adapted accordingly.

In some cases, information gathered during problem determination might produce ideas for new projects for improvement. These ideas should be communicated to the right department(s) in your organization, so that they can be evaluated and do not get lost.

It should be noted that during your career, you will experience some problems that occur only once, and even with the best diagnostic data, cannot be recreated or solved by anyone. When this happens, there is a point in time where you must accept the fact that this anomaly was, in fact, just that: an anomaly.

2.3 Guidelines for customers

In order to understand and resolve your software support service request in the most expedient way possible, it is important that you take the following steps before you contact a software support center. You will need to gather information about the problem and have it on hand when discussing the situation with the software specialist. The following steps are an example of what is required.

2.3.1 Defining the problem

Being able to articulate the problem and symptoms before contacting software support will expedite the problem solving process. It is very important that you be as specific as possible in explaining a problem or question to our software specialists. Our specialists want to be sure that they provide you with exactly the right solution, so the better they understand your specific problem scenario, the better they are able to resolve it. To assist you with problem identification, refer to “Identifying the problem” on page 17.

2.3.2 Gathering background information

To effectively and efficiently solve a problem, the software specialist needs to have all of the relevant information about the problem. Being able to answer the following questions will help in our efforts to resolve your software problem:

- ▶ What levels of software were you running when the problem occurred? Please include all relevant products, that is, operating system as well as related products.
- ▶ Has the problem happened before, or is this an isolated problem?
- ▶ What steps led to the failure?
- ▶ Can the problem be recreated? If so, what steps are required?
- ▶ Have any changes been made to any component of the system?
- ▶ Were any messages or other diagnostic information produced? If so, what were they?

It is often helpful to have a printout of the message number(s) of any messages received when you place the call for support.

Define your technical question in specific terms and provide the version and release level of the product(s) in question.

2.3.3 Gathering relevant diagnostic information

It is often necessary that our software support specialists analyze specific diagnostic information, such as logs, traces, storage dumps, etc., in order to resolve your problem. Gathering this information is often the most critical step in resolving your problem. Product-specific diagnostic documentation can be very helpful in identifying what information is typically required to resolve problems. If you are unsure about what information is required, you can always contact software support for assistance in gathering the needed diagnostic information.

2.3.4 Determining problem severity

When you need to report a problem to the IBM Support Center, you will be asked what the severity of the problem is. We set severity from severity 1 (highest severity, meaning worst problems) to severity 4 (lowest severity, meaning least important problems). It is important that you be realistic when reporting the severity of an issue, so we can prioritize it properly.

Severity 1

Very serious problem that has critical business impact; the production system is down, you are unable to use the product in a production environment, no workaround is available.

Severity 2

Serious problem that has a significant business impact; use of the product is severely limited, but no production system is continuously down. Severity 2 problems include situations where customers are forced to restart processes frequently, and performance problems that cause significant degradation of service but do not render the product totally unusable. In general, a very serious problem for which there is an unattractive, but functional workaround would be severity 2, not severity 1.

Severity 3

Problems that cause some business impact but that can be reasonably circumvented; these are situations where there is a problem but the product is still usable. For example, these could be short-lived problems or problems with components that have failed and then recovered and are back in normal operation at the time the problem is being reported. The default severity of new problem reports should be severity 3.

Severity 4

Minor problems that have minimal business impact.

While we are all aware of the pressure that customers and management place on the speedy resolution of their problem, determining the correct problem severity enables all involved support teams to react and manage the problems with respect to the real severity of the problem. While "customer is unhappy: severity 1" is in many cases valid for business reasons, it does not preclude the fact that if a customer has "production system down: severity 1," this is more important.

2.3.5 Gathering additional information

When speaking with a software support specialist, you should also mention the following items if they apply to your situation:

- ▶ You are under business deadline pressure.
- ▶ Your availability (that is: when you will be able to work with IBM Software Support).
- ▶ You can be reached at more than one phone number.
- ▶ You can designate a knowledgeable alternate contact with whom we can speak.
- ▶ You have other open problems (PMRs/incidents) with IBM regarding this service request.
- ▶ You are participating in an early support program.
- ▶ You have researched this situation prior to calling IBM and have detailed information or documentation to provide for the problem.

2.3.6 Reporting a software problem

IBM does not warrant that our products are defect-free, but we do endeavor to fix them to work as designed. You may be surprised to learn that you play a key role in this effort. Our remote software support is available to provide you with assistance and guidance, but we assume that you will provide information about your system and the failing component, information that is key to resolving the problem.

Dealing with information gathering includes capturing documentation at the time of a failure, applying a trap or trace code to your system, possibly formatting the output from the trap or trace, and sending documentation or trace information, in hardcopy or soft copy, to the remote support center. You are also responsible for obtaining fixes, by downloading or by receiving ones that have been shipped to you on media, applying the fixes to your systems and testing the fixes to ensure they meet your needs. Occasionally, removal of installed fixes may be necessary in the process of isolating problems. Sometimes, fixing a problem will mean the

installation of a later release of the software since some fixes cannot be retrofitted into earlier code.

You need to be aware of your responsibilities when working with an IBM Support Center. If you do not have the required skill or are unwilling to do the work, you can engage a services provider such as IBM Global Services (IGS) or a business partner to assist you, for an additional fee. If you are involved in a services engagement in which IGS or a Business Partner is designing and implementing an application for you, you should insist the statement of work be very clear as to whose responsibility it is to work on suspected code defect issues with IBM, to ensure proper entitlement for remote support.

2.3.7 Accessing Software Support

When submitting a problem to IBM Software Support or calling about a particular service request, please have the following information ready:

- ▶ IBM Customer Number.
- ▶ The machine type/model/serial number (for software maintenance calls also).
- ▶ When purchasing new products under the Passport Advantage® Agreement, you will need your IBM Customer Number as well.
- ▶ Company name.
- ▶ Contact name.
- ▶ Preferred means of contact (voice or e-mail).
- ▶ Telephone number where you can be reached if contact request is voice.
- ▶ Related product and version information.
- ▶ Related operating system and database information.
- ▶ Detailed description of the issue.
- ▶ Severity of the issue in relationship to the impact on your business needs.

Internet access

Through the electronic problem submission Web site(s), you may post support questions electronically to the same support specialists who staff IBM's telephone support lines. Prior to submitting a problem via the Internet, you will need the same information as if you were presenting a problem by telephone. This capability allows you to put all of the pertinent information about your problem into the problem record via the Internet without having to wait for someone to call you back. This should save you time and help with problem resolution time.

If you are submitting a severity 1 problem and it is outside of normal business hours in your country then it may be prudent to follow up with a call to your local support center referencing the problem number you receive on the Web. We want to ensure that your emergency call will be handled in the appropriate time frame.

Software Maintenance offering

Electronic Incident/PMR submission is available with the Software Maintenance offering.

Access to the new tool is available through the IBM Software Support Web site at <http://www.ibm.com/software/support>. Click the box that says **Contact Support**. You must be a registered user/authorized caller to use this tool. Please work with your Site Technical Contact to be authorized for this service capability.

Support Line offering

If you have a current IBM ServiceLink user ID and password, you will still be able to submit problems to IBM through your normal processes on the IBM support Web page at <http://www.ibm.com/support>.

Please note that IBM is not responsible for delays caused by networking problems encountered when reporting problems electronically. Customers are also responsible for acquiring their own Internet service provider should they wish to use either of the Internet submission capabilities listed above.

Voice access

IBM Voice Support is available for all current support contract holders through a Single Point of Contact (SPOC) telephone number in your country (where available). You will be required to provide your IBM Customer Number for validation of the support service to which you are entitled as well as the product about which you are calling. Please refer to <http://techsupport.services.ibm.com/guides/contacts.html> for specific country phone numbers.



The eMerge business scenario

We will use a fictional business scenario in the remaining chapters of this book, onto which we can map the different products and problems. This *eMerge* scenario was originally developed by the IBM WebSphere System House. While it is a fictional scenario, it is based on real-world company merger situations and was developed specifically to show possible integration approaches. We have modified the scenario for our purposes, mainly by simplifying it and leaving out the parts which are irrelevant to this book.

Based on this scenario, we will show the big picture aspects of a solution integrating different IBM WebSphere software products, along with some typical problems in the interaction of these products and their symptoms. Therefore, this chapter can be used as an entry point for problem determination, leading to the correct product-related chapter covering further problem analysis once the general problem area has been identified.

Throughout this chapter, we will reference the Patterns for e-business. The Patterns for e-business capture and publish e-business artifacts that have been used, tested and proven in successful e-business solutions. If you are not familiar with the Patterns for e-business, you can find out more on the Patterns for e-business Web site at:

<http://www.ibm.com/developerWorks/patterns/>

The Patterns used in this chapter can be found in the IBM Redbook *Self-Service Applications using IBM WebSphere V5.0 and IBM MQSeries Integrator*, SG24-6875.

3.1 The eMerge business scenario

The eMerge business scenario is based on the merger of two motor insurance companies.

According to the scenario, Lord General Insurance (LGI) has acquired DirektCar.com to achieve a quick entry into the e-business direct insurance market and to utilize DirektCar.com's existing skills and infrastructure based on Internet and Web technologies. The merged company has chosen the name LGDirect Insurance for its newly combined direct channel.

Lord General Insurance has been in the insurance business for 50 years, focusing on motor and general insurance and has five million policy holders, who are served via traditional channels through agents and an independent call center. Lord General Insurance's information technology infrastructure has grown over the years and is largely based on IBM S/390® and zSeries® technology.

DirektCar.com is a fast expanding dot.com business focused on selling motor insurance over the Internet, with currently fewer than one million policy holders. Their relatively new infrastructure is based on IBM AIX and Microsoft® Windows® 2000 systems.

The merger between the two companies has caused them some difficulties:

- ▶ The two merged companies have customer and policy data of different formats, which cannot be integrated because Lord General Insurance may sell DirektCar.com again in the future.
- ▶ Business logic and business rules are embedded into current applications, which reduces flexibility for the development of new applications utilizing the same business logic and business rules.
- ▶ There are high costs in managing two separate information technology infrastructures.

For the merger to be successful, the companies have decided to integrate their systems while maintaining the direct Internet channel of DirektCar.com, and gradually extending its availability to cover all products and customers within a one-year time frame, complementing the existing agent and call-center channels. For this, they used the Composite pattern *Account Access*, which includes the Business pattern *Self-Service (User-to-Business)* paired with the Integration patterns *Access Integration* and *Application Integration*. The Self-Service pattern is used for applications where users interact with a business via the Internet or an intranet. Access Integration includes the integration of a number of services through a common entry point, and Application Integration defines the

integration of multiple applications and data sources without the user directly invoking them.

A future goal would be to extend the choice of channels available for customers, agents and other third parties.

3.2 Architecture

Lord General Insurance (LGI) and DirektCar.com have merged their formerly separate system and application architectures, with the exception of the back-end systems, which will continue to be used in parallel. IBM WebSphere products were used for the new components of the integrated infrastructure. The application in the scenario is a distributed Java 2 Platform Enterprise Edition (J2EE) application. Figure 3-1 shows an overview of the combined Operational Model. Firewalls shield the LGDirect Insurance systems from the Internet and the Lord General Insurance and DirektCar.com back-end systems.

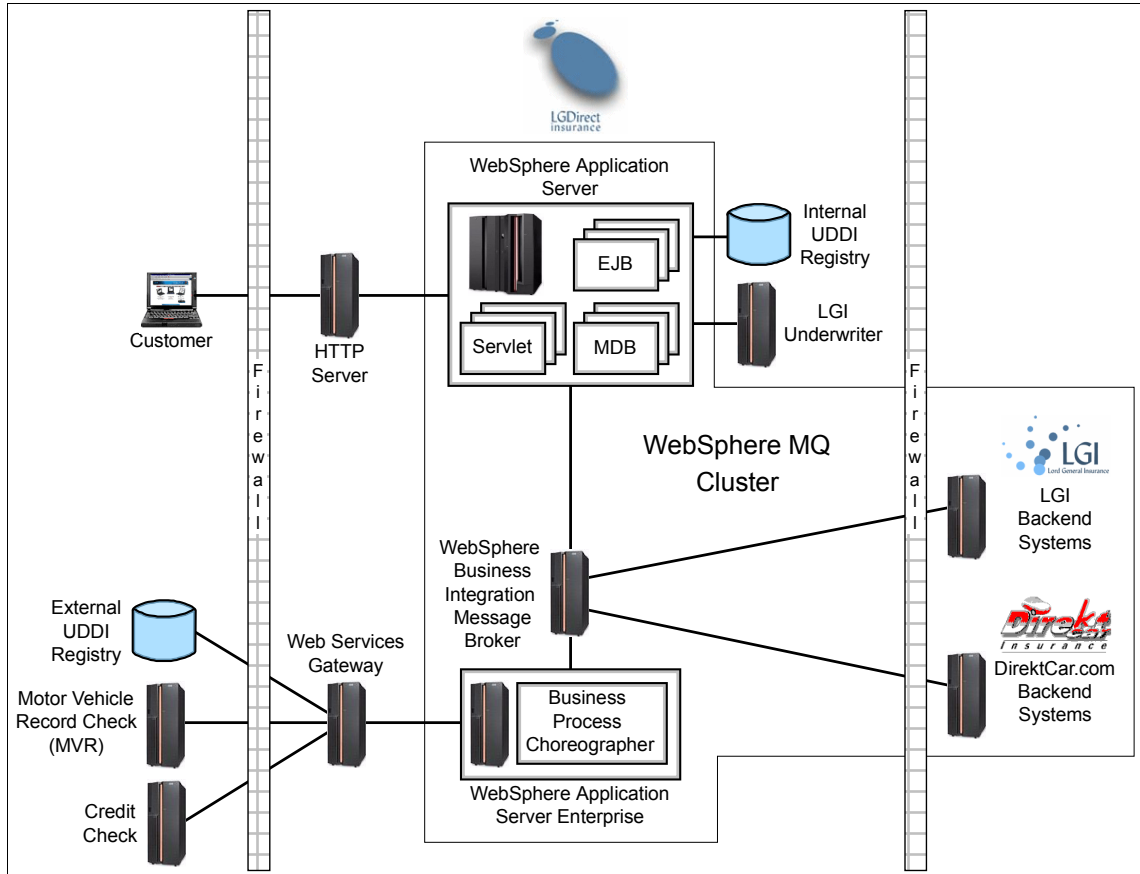


Figure 3-1 Basic Operational Model of the eMerge business scenario

The customer can access the company Web site with a Web browser. The request is served by an HTTP Server running the WebSphere plug-in. The plug-in will route all requests for non-static content to the WebSphere Application Server.

The application components deployed on WebSphere Application Server communicate with the Internal UDDI Registry and the LGI Underwriter Web Service. There is also message-based communication with WebSphere Application Server Enterprise through the WebSphere Business Integration Message Broker. These messages are sent and received via a WebSphere MQ Cluster as a central communication component in the architecture.

WebSphere Application Server Enterprise runs the Business Process Choreographer and controls the communication with external Web services. To protect the internal network during this communication, the Web Services

Gateway is used. It is the Web Services Gateway's responsibility to determine the actual location of the Web service by looking up its description (in the form of a Web Services Description Language (WSDL) document) in the External UDDI Registry. The external Web services used are the Motor Vehicle Record check (MVR) and Credit check.

WebSphere Business Integration Message Broker is used for communication between the back-end systems, WebSphere Application Server and WebSphere Application Server Enterprise. It automatically converts the messages to and from the formats the different back-end systems are expecting. The rules for this conversion are defined in message flows.

As mentioned above, separate back-end systems exist for each of the companies. The LGI back-end system and the DirektCar.com back-end system do not communicate directly with each other.

3.3 Application details

We will not describe the full business functionality of the LGDirect Insurance application, but limit ourselves to two use cases in the scenario, *Get Customer Details* and *Request and Accept Quote*.

3.3.1 Get Customer Details use case

The *Get Customer Details* use case allows an existing customer who has previously registered with LGDirect Insurance to log in to the Web site and retrieve the details which he/she has entered. If the user so chooses, the details can also be changed.

Successful outcome

Based on information entered by the user, the correct customer details are retrieved and displayed by the system. The system offers the user the option to request a quote.

Failure outcome

The customer details cannot be found or displayed by the system.

Main use case scenario

The main scenario is described step by step with the current actor named at the beginning of each step. A successful outcome is implied in the description.

1. Customer: makes a request to access the car quotation function.

2. System: asks if the customer is a new user or returning customer.
3. Customer: enters the company code (LGI for Lord General Insurance or DC for DirektCar.com) and customer number.
4. System: displays known customer details and two options to choose from: details are correct/details are incorrect. The details include:
 - a. First and last name
 - b. Date of birth
 - c. Street name and number
 - d. Post Code (ZIP)
 - e. E-mail address
5. Customer: chooses either **Details are correct** (the use case continues with step 8) or **Details are incorrect**.
6. System: displays the customer details form with input fields pre-filled with known customer details.
7. Customer: changes or adds details. Continue with step 4.
8. Use case ends here.

3.3.2 Request and Accept Quote use case

The *Request and Accept Quote* use case enables a prospective customer to receive and accept a real-time quote for car insurance based on data entered through the Web interface. After acceptance, the quote is issued as a policy, pending final underwriting review and approval.

Note: A successful outcome of the *Get Customer Details* use case is a prerequisite for this use case.

Successful outcome

The customer has answered all the required questions and has submitted the request for a quote. The quote is returned with a value correctly reflecting the risk. The customer has received a real-time quote (that is, in the same Web session). The customer has accepted the quote. The policy will be issued pending further checks.

Failure outcome

LGDirekt Insurance is not able to offer a real-time quote and accept it because:

- ▶ System limitations prohibit it. For example, the system is down or unable to handle a direct quote.
- ▶ The quote is returned with an incorrect quote valuation.

A confirmation message containing a policy reference is provided unless the back-end systems are down or the session ends before the prospect has finished submitting the data. The message will display the reason for failure and direct the prospect to the customer call center for further information.

Main use case scenario

The main scenario is described step by step with the current actor named at the beginning of each step. A successful outcome is implied in the description.

1. Customer: has identified himself via the *Get Customer Details* use case and indicates that he wants a new car quote.
2. System: displays motor details form with input fields for:
 - a. Car manufacturer
 - b. Model
 - c. Value
 - d. Registration
 - e. Color
 - f. Engine size
 - g. Year of manufacture
3. Customer: enters the necessary information.
4. System: checks to see that all details are correct and complete; if not, continues with step 2. Performs a risk check for the prospect against the driver and vehicle selected and calculates a premium.
 - a. If the risk check is successful, displays the quote with the estimated premiums and prompts customer to either accept the quote or store it for later processing (which is another use case).
 - b. If the risk check is unsuccessful, displays a reason for rejection or an error message (for example, system not available). The use case ends here.
5. Customer: accepts the quote.
6. System: displays payment details form with input fields for:
 - a. Credit card number
 - b. Valid from date
 - c. Expiration date
7. Customer: enters requested payment details and requests to purchase the policy.
8. System: displays message indicating that the purchase was successful and that the policy has been provisionally accepted.
9. Customer: ends the session at this point.

10. System: Performs MVR check using the information gathered. Also performs a credit check using name, birth date, and address with postal code. Upon completion, sends an e-mail to the customer with a summary of the policy agreement containing:
 - a. Operating company and policy number
 - b. Issue/effective date
 - c. Total premium
 - d. Payment details
 - e. Terms and conditions
11. The use case ends here.

3.4 The Patterns for e-business approach

LGDirect's application is separated into multiple parts deployed on different server nodes. Most parts of the application, apart from the back-end components, come in the Enterprise Archive (EAR) format defined in the J2EE standard. Others take the form of process flows or message flows. Some static Web content is deployed on the HTTP Server, but other static content is included directly in the EAR files.

The application parts deployed on WebSphere Application Server follow the Model-View-Controller (MVC) design principle. The Controller is implemented as servlets and the Model is implemented as Enterprise Java Beans (EJB). The View is implemented as Java Server Pages (JSP).

The chosen Application pattern is the *Self Service::Decomposition pattern*, which takes a single, compound request from a client, decomposes it into multiple simpler requests and intelligently routes them to multiple back-end applications. Typically, the responses from these multiple back-end applications are recomposed into a single response and sent back to the client. The business logic exists both on the back end and on the middle tier.

The communication with the back end can be synchronous (meaning a response is expected in the same session) or asynchronous (meaning a response can be received after the session has ended). In our scenario, the request for a quote triggers a synchronous request, whereas the quote acceptance triggers both synchronous and then asynchronous (for example, the MVR check) requests.

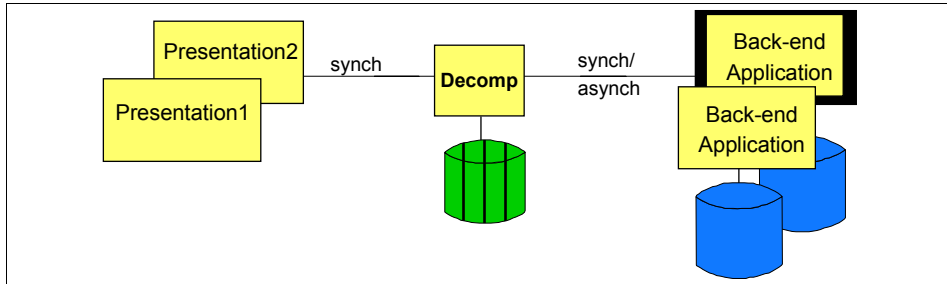


Figure 3-2 Self-Service Decomposition application pattern

Choosing an Application pattern lead to the choice of a Runtime pattern that most closely matches the requirements of the application. A Runtime pattern uses nodes to group functional and operational components. The nodes are interconnected in the architecture. The basic Runtime pattern for the Decomposition pattern is depicted in Figure 3-3 on page 36, with the Decomposition pattern mapped to it.

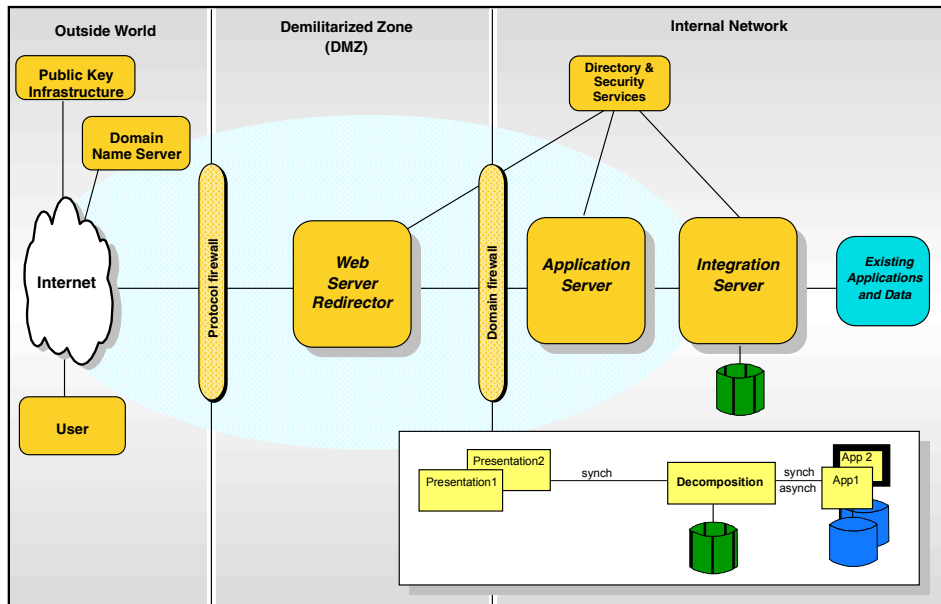


Figure 3-3 Basic Runtime pattern for the Decomposition application pattern

For reasons of simplicity in our examples, we have left out some components from the Runtime pattern in the eMerge scenario, notably the Public Key Infrastructure and the firewalls.

Request processing begins at the Web Server Redirector component, which consists of the HTTP Server and the WebSphere plug-in in our scenario.

Upon receiving a request from the customer's browser, the WebSphere plug-in installed in the HTTP Server sends this request to a servlet deployed on WebSphere Application Server. The handling servlet processes the information in the request and calls the appropriate EJBs to perform the business logic. The EJBs in turn call message flows in the WebSphere Business Integration Message Broker and Business Process Choreographer process flows and back-end functionality to perform their work. Both WebSphere Business Integration Message Broker and Business Process Choreographer take the role of the Integration Server component in the Runtime Pattern.

The LGI Underwriter Web service is called by the message flows via Message Driven Beans (MDB) deployed on WebSphere Application Server. By using an Internal UDDI Registry to retrieve a WSDL document which describes the Web service, the actual location can be changed without affecting other parts of the application.

Communication with the Business Process Choreographer process flows and the back-end systems is done via a WebSphere MQ Cluster, which reliably routes messages to the correct receiver or stores them temporarily for later retrieval.

Because two different back-end systems exist, the messages must be routed to the correct one and converted to the format expected by that back-end system. The other parts of the application are not and should not be aware of this requirement, but instead use a common format for the messages they send. The conversion and routing tasks are performed by message flows in WebSphere Business Integration Message Broker. When the back-end system answers the request, a message flow again converts the answer message to the common format and uses the WebSphere MQ Cluster to send it to the business logic on the WebSphere Application Server, which then performs further processing.

External Web services provide the Motor Vehicle Record (MVR) check and the Credit check. Since these checks together actually form a business process, they are orchestrated by a process flow in the Business Process Choreographer component of WebSphere Application Server Enterprise. Business Process Choreographer executes pre-defined process flows called by other parts of the application. The actual locations of the MVR and Credit check Web services are looked up in a WSDL document retrieved from an External UDDI Registry. Since the External UDDI Registry and these two Web services are external to LGDirect, the process flows do not communicate directly with them, but through the Web Services Gateway.

When the handling servlet receives the final result of the request from the business logic, it hands over control to the appropriate JSP. The JSP then uses the information passed to it by the servlet to send an HTTP response to the customer's browser.

3.5 Products

There are a numerous products involved in any IT solution; our eMerge scenario is not different in this sense. The solution includes several IBM products and applications.

3.5.1 Web server

The Web server is a core module for providing Web content, mostly static content. The Web server in our scenario is the IBM HTTP Server from the WebSphere Application Server installation.

Find information about IBM HTTP Server problem determination at:

<http://www-306.ibm.com/software/webservers/httpservers/support/>

Plug-in

The Web server plug-in that is part of the WebSphere Application Server is integrated into the Web server. This plug-in is responsible for routing the client requests to the application server whenever necessary.

3.5.2 WebSphere Application Server

The main middleware engine in the eMerge scenario is WebSphere Application Server, which is a J2EE application server implementation from IBM. WebSphere Application Server has different editions, two of which were used in the solution.

WebSphere Application Server base

The base application server is a J2EE certified server providing all the basic functionality for a middleware application.

WebSphere Application Server Enterprise

Some applications have further requirements for the runtime environment and special features not specified for the base J2EE application server. WebSphere Application Server Enterprise extends the J2EE specification and programming model with useful features like Process Choreographer, Asynchronous Beans, Extended Messaging, and many more.

Find information about WebSphere Application Server Enterprise problem determination at:

<http://www-306.ibm.com/software/webservers/appserv/was/support/>

3.5.3 Message Oriented Middleware

Message Oriented Middleware is the glue in the eMerge scenario that keeps the application components together. It provides asynchronous messaging between front-end and back-end servers.

WebSphere MQ

WebSphere MQ is the Message Oriented Middleware provider for the eMerge scenario. WebSphere MQ provides asynchronous messaging for the application components together with many basic integration functions.

Find information about WebSphere MQ problem determination at:

<http://www-306.ibm.com/software/integration/wmq/support/>

3.5.4 Integration middleware

The integration middleware is a component that provides integration features and capabilities for application components. This middleware component's primary function is to integrate the otherwise separate or non-interoperable components.

WebSphere Business Integration Message Broker

WebSphere Business Integration Message Broker is IBM's integration middleware solution built on top of WebSphere MQ. The message broker operates over asynchronous messaging and connects the systems through message queues. Messages can be decomposed, transformed, recomposed, routed, and persisted as requested in order to integrate components via messaging.

Find information about WebSphere Business Integration Message Broker problem determination at:

<http://www-306.ibm.com/software/integration/wbimessagebroker/support/>

3.5.5 Back-end applications

Back-end applications are not the focus of this book, but they are an integral part of the eMerge scenario.

DB2 Universal Database

The IBM DB2 Universal Database is the primary database for the eMerge scenario. It provides relational database storage for the applications and supporting databases for the application components.

Find information about IBM DB2 UDB problem determination at:

<http://www-306.ibm.com/software/data/db2/udb/support.html> and
<http://www-106.ibm.com/developerworks/db2/library/techarticle/0205bargas/0205bargas.html>

Enterprise applications

The back end will come into view as the possible location of the problem cause when you have ruled out the other components with the methods outlined above. This does not imply that the back end is more seldom the cause of problems than the other components. However, in the eMerge scenario, the communication with the back end is done via WebSphere MQ and WebSphere Business Integration Message Broker, and we have covered these in this chapter already. The same techniques can be applied when trying to narrow down the problem in the area of back-end communication, so we will not repeat them here.

We treat the back-end applications as a “black box” in most parts of the book. However, we cover some problem determination techniques for back-end applications commonly used in e-business solutions in Chapter 12, “The back-end diagnostic overview” on page 301. You can find more information in that chapter.



Part 2

Problem determination details



Problem determination tools: runtime environment

The purpose of this chapter is to provide some guidelines to assist with the collection of relevant diagnostic data to be used for problem diagnosis and resolution.

This chapter does not offer a definitive list of techniques, but covers most of the key diagnostic tools available for capturing and analyzing data in WebSphere Application Server, WebSphere Business Integrator and WebSphere MQ environments. It is not implied that the techniques outlined will guarantee that the data collected will be sufficient to solve the reported problem, nor is it excluded that additional data will be requested to further the problem diagnosis process.

This chapter describes some basic fundamentals to diagnose system problems. There are times, however, when your diagnosis will require the assistance of the IBM Support Center.

You can also start your research for problem determination using Internet search engines to look for known issues documented somewhere on the Web or in a newsgroup. One of the search engines you may find useful is:

<http://www.google.com>

4.1 Chapter organization

This chapter includes problem determination tools for operating systems and the products that we cover in this book.

You will first find information about which release you are running and details about the operating system (AIX).

We then cover logs and problem determination tools for the major and related products that are part of the eMerge scenario. These includes:

- ▶ WebSphere Application Server
- ▶ WebSphere MQ
- ▶ WebSphere Business Integrator
- ▶ IBM HTTP Server
- ▶ IBM Directory Server
- ▶ IBM DB2 UDB
- ▶ These products running with z/OS

These tools and the log files are essential to problem determination. The product-specific chapters will refer to these tools and logs. Refer to the tools and logs here for detailed information on finding, using and understanding them.

4.2 Operating system

Version: AIX 5.2 ML 2

There are specific tools that you can use as problem determination tools, depending on your operating system. They may help you view system logs or monitor processes.

What release am I running?

Different platforms use different commands to show you product information.

Important: Do not overlook the most obvious source of release information, which is often recorded in the console or job log messages generated during startup of the operating system or product.

To determine your operating system version, use the **oslevel** command.

```
# oslevel  
5.2.0.0
```

You can also determine release levels for certain packages or products using the **ls1pp** command; for example, **ls1pp -l all** lists all the installed packages. You can also filter between the packages; for example, **ls1pp -l | grep db2** will give you the DB2 package version.

Tracing in AIX

AIX has a system call tracing facility, but it is difficult to use since it cannot be directed at a single process; turning on tracing will cause the system to trace all calls made by all processes on the machine, and you will not be able to tell which process made which call from the trace output. The relevant commands are **trace**, **trcstop**, and **trcrpt**; see the AIX documentation for usage information.

tcpdump and iptrace

The AIX **tcpdump** command is similar to Solaris' snoop (if you are familiar with that environment). Alternatively, the combination of **iptrace** and **ipreport** can be used on AIX. See the AIX documentation for details; in particular, you can try **man tcpdump** and **man iptrace**.

To trace all packets going between machines box1 and box2 using **tcpdump**, use:

```
tcpdump -x host box1 and box2
```

UNIX® system core dump analysis

UNIX processes (including the JVM process) will produce a system core dump as well as Java stack trace information in a process's working directory if it crashes. The system core dump can provide useful information as to why the process crashed, giving you a system view of a failing JVM process. However, the system core dump will not provide Java class information. Everything in the dump is C library oriented. The information provided for the JVM process refers to Java's C libraries and not the reference Java class files.

Looking at a system core dump

The core file on UNIX systems can be inspected using the **dbx** and **gdb** (GNU debugger) tools. The **dbx** tool is part of the AIX install (it might not be installed by default).

You can issue the **dbx** command with the Java binary executable file, normally `<WebSphere_home>/java/bin/java`, as the parameter. The commands listed in Example 4-1 on page 46 show how to find the binary executable and invoke **dbx**.

Example 4-1 Invoking dbx

```
# pwd
/usr/WebSphere/AppServer
# ls -alF core
-rw-r--r--  1 root system 425440811 Nov 07 19:40 core
# dbx /usr/WebSphere/AppServer/java/bin/java
Type 'help' for help.
reading symbolic information ...warning: no source compiled with -g
[using memory image in core]
Segmentation fault in strncpy.strncpy [/usr/lib/libbadjni.so] at 0xd32a3318
0xd32a3318 (strncpy+0x118) 9cc50001      stbu   r6,0x1(r5)
(dbx)
```

The sample was taken from AIX 4.3.3 and AIX 5.1 systems. It shows that a segmentation fault happened.

If you do not know where the Java binary is located, the following command will display the true Java executable name of the core:

```
# strings core | grep COMMAND_LINE
```

or

```
# strings core | more
```

After you start **dbx**, the **where** command provides a stack trace of where the error occurred, as shown in Example 4-2.

Example 4-2 dbx where command

```
(dbx) where
strncpy.strncpy() at 0xd32a3318
bad_native_method() at 0xd32a315c
Java_itso_BadJni_badJniMethod(0x44b1fe4c, 0x4574c2e0) at 0xd32a31ac
mmisInvoke_V_VHelper(0x32c13ef0, 0x44b235fc, 0x1, 0x44b1fe4c, 0x4574c358) at
0xd2eb6fe4
mmipInvoke_V_V(??, ??) at 0xd2ed5e6c
```

The culprit is to be found in the `bad_native_method()` method of the native JNI library module. The **registers** and **listi** commands are also useful to view the system registers information and instructions.

Example 4-3 Analyzing system core dump with dbx

```
(dbx) unset $noflregs
(dbx) registers
  $r0:0x00000000  $stkp:0x4574c1c8  $toc:0x494a13c4  $r3:0x00000000
  $r4:0x4574c20b  $r5:0xffffffff  $r6:0x00000045  $r7:0x00000074
  $r8:0x000000c2  $r9:0x00000058  $r10:0x40b55a74  $r11:0x000034e0
```



```

$r12:0x00000000 $r13:0x00000000 $r14:0x456cf800 $r15:0x4574c350
$r16:0x44b1fe4c $r17:0x40f0b1ec $r18:0x00000009 $r19:0x00000000
$r20:0x00000041 $r21:0x5604a124 $r22:0x000000c1 $r23:0x00000001
$r24:0x32451780 $r25:0x40b20600 $r26:0x44b235fc $r27:0x30213b08
$r28:0x3021c118 $r29:0x44b222b4 $r30:0x44b1fe4c $r31:0x30212418
$iar:0xd32a3318 $msr:0x0000d0b2 $cr:0x44824844 $link:0xd32a3160
$ctr:0x0000000a $xer:0x00000000

Condition status = 0:g 1:g 2:l 3:e 4:g 5:l 6:g 7:g
$fr0:0x0000000000000000 $fr1:0x3fe8000000000000 $fr2: 0x0000000000000000
$fr3:0x0000000000000000 $fr4:0x0000000000000000 $fr5: 0x0000000000000000
$fr6:0x0000000000000000 $fr7:0x0000000000000000 $fr8: 0x0000000000000000
$fr9:0x0000000000000000 $fr10:0x0000000000000000 $fr11: 0x0000000000000000
$fr12:0x0000000000000000 $fr13:0x3fe8000000000000 $fr14: 0x0000000000000000
$fr15:0x0000000000000000 $fr16:0x0000000000000000 $fr17: 0x0000000000000000
$fr18:0x0000000000000000 $fr19:0x0000000000000000 $fr20: 0x0000000000000000
$fr21:0x0000000000000000 $fr22:0x0000000000000000 $fr23: 0x0000000000000000
$fr24:0x0000000000000000 $fr25:0x0000000000000000 $fr26: 0x0000000000000000
$fr27:0x0000000000000000 $fr28:0x0000000000000000 $fr29: 0x0000000000000000
$fr30:0x0000000000000000 $fr31:0x0000000000000000 $fpscr: 0xa6100000
in strncpy.strncpy [/usr/lib/libbadjni.so] at 0xd32a3318
0xd32a3318 (strncpy+0x118) 9cc50001 stbu r6,0x1(r5)

```

```

(dbx) listi .-40,+.40
0xd32a32f4 (strncpy+0xf4) 4182009c beq 0xd32a3390 (strncpy+0x190)
0xd32a32f8 (strncpy+0xf8) 8cc40001 lbzu r6,0x1(r4)
0xd32a32fc (strncpy+0xfc) 8ce40001 lbzu r7,0x1(r4)
0xd32a3300 (strncpy+0x100) 8d040001 lbzu r8,0x1(r4)
0xd32a3304 (strncpy+0x104) 8d240001 lbzu r9,0x1(r4)
0xd32a3308 (strncpy+0x108) 2c060000 cmpi cr0,0x0,r6,0x0
0xd32a330c (strncpy+0x10c) 2c870000 cmpi cr1,0x0,r7,0x0
0xd32a3310 (strncpy+0x110) 2f080000 cmpi cr6,0x0,r8,0x0
0xd32a3314 (strncpy+0x114) 2f890000 cmpi cr7,0x0,r9,0x0
0xd32a3318 (strncpy+0x118) 9cc50001 stbu r6,0x1(r5)
0xd32a331c (strncpy+0x11c) 4e400020 bdzgelr
0xd32a3320 (strncpy+0x120) 4182004c beq 0xd32a336c (strncpy+0x16c)
0xd32a3324 (strncpy+0x124) 9ce50001 stbu r7,0x1(r5)
0xd32a3328 (strncpy+0x128) 4e400020 bdzgelr
0xd32a332c (strncpy+0x12c) 4186004c beq cr1,0xd32a3378 (strncpy+0x178)
0xd32a3330 (strncpy+0x130) 9d050001 stbu r8,0x1(r5)
0xd32a3334 (strncpy+0x134) 4e400020 bdzgelr
0xd32a3338 (strncpy+0x138) 419a004c beq cr6,0xd32a3384 (strncpy+0x184)
0xd32a333c (strncpy+0x13c) 9d250001 stbu r9,0x1(r5)
0xd32a3340 (strncpy+0x140) 4e400020 bdzgelr
0xd32a3344 (strncpy+0x144) 419e004c beq cr7,0xd32a3390 (strncpy+0x190)
(dbx) quit

```

More useful commands of **dbx** are:

- ▶ **map**
- ▶ **thread**
- ▶ **thread info**
- ▶ **thread <thread_no>**
- ▶ **thread current <thread_no>**

Type **help** to access help for a command or topic and **quit** to exit **dbx**.

How to make sure that a good core file is generated

If, after a crash, there is no core file or the output from **dbx** shows that the core file is truncated, check the following:

1. Ensure that the file system containing the core file has enough free space. The size required depends on your WebSphere configuration environment, but it is recommended that you have over 500 MB.
2. On AIX, ensure that *Enable full CORE dump* is set to true in the system environment. You can do this by using **smitty**:

```
# smitty chgsys
```

or by using the following command:

```
# lsattr -El sys0 | grep fullcore
fullcore      false          Enable full CORE dump  True
# chdev -a fullcore=true -l sys0
sys0 changed
# lsattr -El sys0 | grep fullcore
fullcore      true           Enable full CORE dump  True
```

3. The owner of the running process must have write permission to the directory where the process dumps the core.
4. Both of the maximum file and coredump size specifications must be large enough.

```
# ulimit -a
time(seconds)      unlimited
file(blocks)      2097151 <-- !
data(kbytes)       131072
stack(kbytes)      32768
memory(kbytes)     32768
coredump(blocks)  2097151 <-- !
nofiles(descriptors) 2000
```

You can change the file and coredump maximum value using the following command:

```
# ulimit -f nnnnnn
# ulimit -c nnnnnn
```

The directory where the process dumps the core is the current working directory of the process that is running. In WebSphere Application Server V5, this is normally the <WebSphere_root>\ directory.

Monitoring a running process with dbx

Another use of **dbx** is to monitor a running process. The **-a** parameter allows the debug program to be attached to a process that is running. To attach the debug program, you need authority to use the **kill** command on this process. Use the **ps** command to determine the process ID. If you have permission, the **dbx** command interrupts the process, determines the full name of the object file, reads in the symbolic information, and prompts for commands.

Example 4-4 Monitoring a running process with dbx

```
# ps -ef|grep java
root 18088      1 0 Nov 07   pts/2   9:06 /usr/WebSphere/... dmgr
root 20648      1 6 Nov 07   pts/2  39:58 /usr/WebSphere/... m10df51f
root 22234 20648 2 Nov 08   pts/2  32:19 /usr/WebSphere/... server1

# dbx -a 22234
Waiting to attach to process 22234 ...
Successfully attached to java.
Type 'help' for help.
reading symbolic information ...warning: no source compiled with -g

stopped in _event_sleep at 0xd00549dc
0xd00549dc (_event_sleep+0x90) 80410014      1wz   r2,0x14(r1)
(dbx) where
_event_sleep(??, ??, ??, ??, ??) at 0xd00549dc
_event_wait(??) at 0xd0054ec8
_cond_wait_local(??, ??, ??) at 0xd0060e04
_cond_wait(??, ??, ??) at 0xd0061298
pthread_cond_wait(??, ??) at 0xd0061fbc
condvarWait(??, ??, ??) at 0xd3056160
sysMonitorWait(??, ??, ??, ??) at 0xd305511c
lkMonitorWait(??, ??, ??, ??) at 0xd2f39724
JVM_MonitorWait(??, ??, ??, ??) at 0xd2eab14c
(dbx) detach
#
```

To continue execution of the application and exit **dbx**, type **detach** instead of **quit** (if you typed **quit** to exit, the process will stop running).

Note: Do not use the command **dbx -a <pid>** on heavily loaded systems because it could be temporarily blocked.

Error reporting with errpt

Often, the most readily available source of data identifies the key piece of information that will resolve the problem, and often, this source of data is overlooked. The **console log**, **system log**, **error log** related to a specific product, or the whole system is the first place to look when reviewing a problem.

While a system dump or trace is often required, the logs will provide enough detail, in many cases, to solve the problem. The location of the relevant logs varies from product to product, and from system to system.

Important: Do not ignore the valuable data that is written to the log files.

On AIX, the **errpt** command generates an error report from entries in a system error log. If you have a system core dump, the event would be logged in the system error log. You can check it as follows.

Example 4-5 errpt command

```
# errpt -a > /tmp/errpt.txt
# vi /tmp/errpt.txt
....
-----
LABEL:          CORE_DUMP
IDENTIFIER:     C60BB505

Date/Time:      Thu Nov  7 19:40:49 EST
...
Detail Data
SIGNAL NUMBER
      11
USER'S PROCESS ID:
      22226
...
PROGRAM NAME
java
ADDITIONAL INFORMATION
strncpy 118
bad_nativ 20
Java_itso 18
mmisInvok 2A4
entryCmp FFFFE688
??
...
```

```
SYMPTOM CODE
PCSS/SPI2 FLDS/java SIG/11 FLDS/strncpy VALU/118 FLDS/bad_nativ
-----
...

```

A portion of the name of the native methods processing at the time of the JVM crash is shown in the error report. Even though the `errpt` command does not provide the fully qualified name, it is enough to start tracing the problem.

Syslog daemon

On UNIX, all WebSphere Business Integration Message Broker messages (other than those generated by the command line utilities) are sent to the syslog, therefore it is useful to redirect user messages to a separate file.

To redirect user messages to a file called `user.log`, follow these steps:

1. Log in as root.
2. Add the following line to the `/etc/syslog.conf` file to redirect debug level messages to the file `user.log`:

```
user.debug /var/adm/user.log
```

You can add similar lines to direct information, warning and error messages to `user.log`.

3. Enter the following commands to create a file called `user.log`:

```
touch /var/adm/user.log
chown root:mqbrkrs /var/adm/user.log
chmod 750 /var/adm/user.log
```

4. Restart the syslog daemon.

On AIX, enter the following command:

```
refresh -s syslogd
```

On HP-UX and Solaris, enter the following command:

```
kill -HUP `cat /etc/syslog.pid`
```

To clear this log, delete `user.log` and repeat steps 3 and 4.

For other syslog options, see the documentation for your particular UNIX platform.

4.3 WebSphere Application Server

Version: WebSphere Application Server Enterprise V5.02.

This section discusses WebSphere Application Server problem determination.

4.3.1 JVM logs

The primary source for JVM diagnostic information for any Java application can be found at:

<http://www-106.ibm.com/developerworks/java/jdk/diagnosis/>

WebSphere Application Server writes system messages to several general purpose logs. These include the JVM logs, the process logs and the IBM service log.

The JVM logs are created by redirecting the stdout and stderr streams of the JVM to independent log files. WebSphere Application Server writes formatted messages to the System.out stream. In addition, applications and other code can write to these streams using the print() and println() methods defined by the streams. Some JDK built-ins such as the printStackTrace() method on the Throwable class can also write to these streams. Typically, the System.out log is used to monitor the health of the running application server. The System.out log can be used for problem determination, but it is recommended that you use the IBM Service log and the advanced capabilities of the Log Analyzer instead. The System.err log contains exception stack trace information that is useful when performing problem analysis.

Since each application server represents a JVM, there is one set of JVM logs for each application server and all of its applications, located by default in the <WebSphere_root>/logs/server_name directory. In the case of a WebSphere Application Server Network Deployment configuration, JVM logs are also created for the deployment manager and each node manager, since they also represent JVMs.

The process logs are created by redirecting the stdout and stderr streams of the process to independent log files. Native code, including the Java Virtual Machine (JVM) itself, writes to these files. As a general rule, WebSphere Application Server does not write to these files. However, these logs can contain information relating to problems in native code or diagnostic information written by the JVM.

As with JVM logs, there is a set of process logs for each application server, since each JVM is an operating system process, and in the case of a WebSphere

Application Server Network Deployment configuration, a set of process logs for the deployment manager and each node manager.

The IBM service log contains both the WebSphere Application Server messages written to the System.out stream and some special messages that contain extended service information that is normally not of interest, but can be important when analyzing problems. There is one service log for all WebSphere Application Server JVMs on a node, including all application servers. The IBM Service log is maintained in a binary format and requires a special tool to view. This viewer, the Log Analyzer, provides additional diagnostic capabilities. In addition, the binary format provides capabilities that are utilized by IBM support organizations.

In addition to these general purpose logs, WebSphere Application Server contains other specialized logs that are very specific in nature and scoped to a particular component or activity. For example, the HTTP Server plug-in maintains a special log. Normally, these logs are not of interest, but you might be instructed to examine one or more of them while going through specific problem determination procedures.

There are also several tools to run which will collect or display information about WebSphere Application Server. These may maybe useful to you or requested by IBM support, such as the collector tool which gathers your log and system information into a jar file.

Reviewing the JVM logs

The JVM logs are written as plain text files. Therefore, there are no special requirements to view these logs. They are located in the <Websphere_root>/logs/applicationServerName directory, and by default are named SystemOut.log and SystemErr.log .

There are two techniques that you can use to view the JVM logs for an application server:

- ▶ Use the Administrative Console. This supports viewing the JVM logs from a remote machine.
- ▶ Use a text editor on the machine where the logs are stored.

The following steps can be used to view the log data.

- ▶ Viewing the JVM logs from the Administrative Console:
 - a. Start the Administrative Console.
 - b. Click **Troubleshooting -> Logging and Tracing** in the console navigation tree, then click **Server -> JVM Logs**.
 - c. Select the **Runtime** tab.

- d. Click the view corresponding to the log you want to view.
- ▶ Viewing the JVM logs from the machine where they are stored:
 - a. Go to the machine where the logs are stored.
 - b. Open the file in a text editor or drag and drop the file into an editing and viewing program.

Interpreting the JVM log data

The JVM logs contain print data written by applications. The application can write this data directly in the form of `System.out.print()`, `System.err.print()`, or other method calls. The application can also write data indirectly by calling a JVM function, such as an `Exception.printStackTrace()`. In addition, the `System.out` JVM log contains system messages written by the WebSphere Application Server.

You can format application data to look like WebSphere Application Server system messages by using the Installed Application Output field of the JVM Logs properties panel, or by using plain text with no additional formatting. WebSphere Application Server system messages are always formatted.

Depending on how the JVM log is configured, formatted messages can be written to the JVM logs in either basic or advanced format.

JVM log message formats

Formatted messages are written to the JVM logs in one of two formats:

- ▶ **Basic format** - The format used in earlier versions of WebSphere Application Server.
- ▶ **Advanced format** - Extends the basic format by adding information about an event, when possible.

Basic and advanced format fields

Basic and advanced formats use many of the same fields and formatting techniques. The various fields that may be found in these formats are as follows:

- ▶ **TimeStamp** - The timestamp is formatted using the locale of the process where it is formatted. It includes a fully qualified date (for example `YYMMDD`), 24 hour time with millisecond precision and a time zone.
- ▶ **ThreadId** - An eight-character hexadecimal value generated from the hash code of the thread that issued the message.
- ▶ **ShortName** - The abbreviated name of the logging component that issued the message or trace event. This is typically the class name for WebSphere Application Server internal components, but can be some other identifier for user applications.

- ▶ **LongName** - The full name of the logging component that issued the message or trace event. This is typically the fully qualified class name for WebSphere Application Server internal components, but can be some other identifier for user applications.
- ▶ **EventType** - A one-character field that indicates the type of the message or trace event. Message types are in upper case. Possible values include:
 - **A** - Audit message.
 - **I** - Informational message.
 - **W** - Warning message.
 - **E** - Error message.
 - **F** - Fatal message.
 - **O** - Message written directly to System.out by the user application or internal components.
 - **R** - Message written directly to System.err by the user application or internal components.
 - **u** - Special message type used by the message logging component of the WebSphere Application Server runtime.
 - **Z** - A placeholder to indicate that the type was not recognized.
- ▶ **ClassName** - The class that issued the message or trace event.
- ▶ **MethodName** - The method that issued the message or trace event.
- ▶ **Organization** - The organization owning the application that issued the message or trace event.
- ▶ **Product** - The product that issued the message or trace event.
- ▶ **Component** - The component within the product that issued the message or trace event.

Process logs

WebSphere Application Server processes contain two output streams that are accessible to native code running in the process. These streams are the *stdout* and *stderr* streams. Native code, including the JVM, may write data to these process streams. In addition, the JVM-provided System.out and System.err streams can be configured to write their data to these streams also.

By default, the stdout and stderr streams are redirected to log files at application server startup; these contain text written to the stdout and stderr streams by native modules (.DLLs, .EXEs, UNIX libraries, and other modules). By default, these files are stored in the <WebSphere_root>/logs/applicationServerName/native_stderr.log and native_stdout.log files.

This is a change from previous versions of WebSphere Application Server, which by default had one log file for both JVM standard output and native standard output, and one log file for both JVM standard error and native error output.

4.3.2 Viewing the service log

The service log is a special log written in binary format. You cannot view the log directly using a text editor. You should never directly edit the service log, because doing so will corrupt the log. To move the service log from one machine to another, you must use a mechanism such as FTP, which supports binary file transfers.

You can view the service log in two ways:

- ▶ It is recommended that you use the Log Analyzer tool to view the service log. This tool provides interactive viewing and an analysis capability that is helpful in identifying problems.
- ▶ If you are unable to use the Log Analyzer tool, you can use the Showlog tool to convert the contents of the service log to a text format which you can then write to a file or dump to the command shell window. The steps for using the Showlog tool are described below.
 - a. Open a shell window on the machine where the service log resides.
 - b. Change the directory to <WebSphere_root>/bin.
 - c. (Optional) Run the **showlog** script with no parameters to display usage instructions. On UNIX systems, the script is named **showlog.sh**.
 - d. Run the **showlog** script with the appropriate parameters.

For example, to display the service log contents to the shell window, use the invocation **showlog service_log_filename**. If the service log is not in the default location, you must fully qualify the **service_log_filename**.

To format and write the service log contents to a file, use the invocation **showlog service_log_filename output_filename**. If the service log is not in the default location, you must fully qualify the **service_log_filename**.

4.3.3 Log Analyzer

To take advantage of the Log Analyzer's browsing and analysis capabilities, start the Log Analyzer tool, **waslogbr**, from the <WebSphere_root>/bin directory.

Click **File -> Open** and select the file **/logs/activity.log**. You can also browse to activity logs from other WebSphere Application Server installations, and even other versions of the product. Expand the tree of administrative and application server logging sessions. Uncolored records are normal records, yellow records

are warnings, and pink records are errors. If you select a record, you will see its contents, including the basic error or warning message, date, time, which component logged the record, and which process (an administrative server or an application server) it came from, in the upper right-hand pane.

The activity log does not analyze any other log files, such as default_stderr.log or tracefile. To analyze these records, right-click a record in the tree on the left (click the **UnitOfWorkView** at the top to select them all), and select **Analyze**. A record with a green check mark next to it matches a record in the symptom database. When you select a check-marked record, you will see an explanation of the problem in the lower right-hand pane.

Updating the symptom database

The database of known problems and resolutions used when you click the **Analyze** menu item is periodically enhanced as new problems come to light and new versions of the product are introduced. To ensure that you have the latest version of the database, use the **File -> Update Database -> WebSphere Application Server Symptom Database** menu item from within the Log Analyzer tool at least once a month. Users who have just installed the product and have never run the update should do so immediately, since updates may have occurred since the version was first released.

The knowledge base used for analysis is built gradually from problems reported by users. In a recently released version of the product, you might not find any analysis hits. However, the Log Analyzer tool still provides a way to see all error messages and warnings from all components in a convenient, user-friendly way.

4.3.4 Version information

When searching for existing problems or working with IBM support, you will need to know your version of WebSphere Application Server. Running **versionInfo** in the WebSphere Application Server bin directory will list the version, build number, fix packs and extensions for your install.

Example 4-6 Running versionInfo on an AIX system

```
# ./versionInfo.sh
WVER0010I: Copyright (c) IBM Corporation 2002; All rights reserved.
WVER0011I: WebSphere Application Server Release 5.0
WVER0012I: VersionInfo reporter version 1.14, dated 5/9/03
```

```
-----
-
IBM WebSphere Application Server Product Installation Status Report
-----
-
```

Report at date and time 2003-12-18T15:54:52-05:00

Installation

-
Product Directory /usr/WebSphere/AppServer
Version Directory \${product.dir}/properties/version
DTD Directory \${version.dir}/dtd
Log Directory /usr/WebSphere/AppServer/logs/update
Backup Directory \${version.dir}/backup
TMP Directory /tmp

Installation Platform

-
Name IBM WebSphere Application Server
Version 5.0

Technology List

-
BASE installed

Installed Product

-
Name IBM WebSphere Application Server
Version 5.0.2
ID BASE
Build Level ptf2M0325.01
Build Date 06/23/2003

-
End Installation Status Report

-
#

4.3.5 Collector tool

The collector tool gathers information about your WebSphere Application Server installation and packages it in a Java archive (JAR) file that you can send to IBM Customer Support to assist in determining and analyzing your problem. Information in the JAR file includes logs, property files, configuration files,

operating system and Java data, and the presence and level of each software prerequisite.

WebSphere Application Server 5.0.2 introduced the `-Summary` option for the collector tool to produce a smaller text file and the console version of some of the information in the Java archive (JAR) file that the tool produces without the `-Summary` parameter. You can use the collector `-Summary` option to retrieve basic configuration and prerequisite software level information.

The collector tool must be used from a directory outside of your application server install. Run the command `<WebSphere_root>/bin/collector.sh`.

The directory where the command was executed will afterwards contain a JAR file named `<Your_System_Name>-BASE-WASenv.jar`.

4.3.6 Debugging with the Application Server Toolkit

The Application Server Toolkit, included with the WebSphere Application Server on a CD which is installable separately, includes debugging functionality that is built on the Eclipse workbench and includes the following:

The *WebSphere Application Server debug adapter* allows you to debug Web objects running on WebSphere Application Server and which you have launched in a browser. These objects include EJBs, JSPs, and servlets.

The *JavaScript debug adapter* enables server-side JavaScript debugging.

The *Compiled language debugger* allows you to detect and diagnose errors in compiled-language applications.

The *Java development tools* (JDT) debugger allows you to debug Java.

All of the debug components in the Application Server Toolkit can be used for local and remote debugging.

4.3.7 WebSphere Application Server tracing

WebSphere Application Server tracing can be used to obtain detailed information about the execution of WebSphere Application Server components, including application servers, clients, and other processes in the environment.

Trace files show the time and sequence of methods called by WebSphere Application Server base classes, and you can use these files to pinpoint the failure.

Collecting a trace is often requested by IBM technical support personnel. If you are not familiar with the internal structure of WebSphere Application Server, the trace output might not be meaningful to you.

Enabling tracing

The trace for an application server process is enabled while the server process runs using the Administrative Console. You can configure the application server to start in a trace-enabled state or to start an immediate trace by setting the appropriate configuration properties. You can only enable trace for an application client or stand-alone process at process startup.

Trace strings

By default, the trace service for all WebSphere Application Server components is disabled. To request a change to the current state of the trace service, a trace string is passed to the trace service. This trace string encodes the information detailing which level of trace to enable or disable and for which components.

You can type in Trace strings, or construct them with a user-friendly GUI in the Administrative Console. Trace strings must conform to a specific grammar for processing by the trace service. The specification of this grammar follows:

```
TRACESTRING=COMPONENT_TRACE_STRING [:COMPONENT_TRACE_STRING ] *
COMPONENT_TRACE_STRING=COMPONENT_NAME=LEVEL=STATE [,LEVEL=STATE ] *
LEVEL =all |entryExit |debug |event
STATE =enabled |disabled
COMPONENT_NAME =COMPONENT |GROUP
```

The `COMPONENT_NAME` is the name of a component or group registered with the trace service. Typically, WebSphere Application Server components register using a fully qualified Java classname, for example `com.ibm.servlet.engine.ServletEngine`. In addition, you can use a wildcard character of asterisk (*) to terminate a component name and indicate multiple classes or packages. For example, use a component name of `com.ibm.servlet.*` to specify all components whose names begin with `com.ibm.servlet`.

Examples of legal trace strings include:

```
com.ibm.ejs.ras.ManagerAdmin=debug=enabled
com.ibm.ejs.ras.ManagerAdmin=all=enabled,event=disabled
com.ibm.ejs.ras.*=all=enabled
com.ibm.ejs.ras.*=all=enabled:com.ibm.ws.ras=debug=enabled,entryexit=enabled
```

Trace strings cannot contain blanks.

Trace strings are processed from left to right. Specifying a trace string like

```
abc.*=all=enabled,event=disabled
```

first enables all traces for all components whose names start with abc, then disables event tracing for those same components. This means that the trace string

```
abc.*=all=enabled,event=disabled
```

is equivalent to

```
abc.*=debug=enabled,entryexit=enabled
```

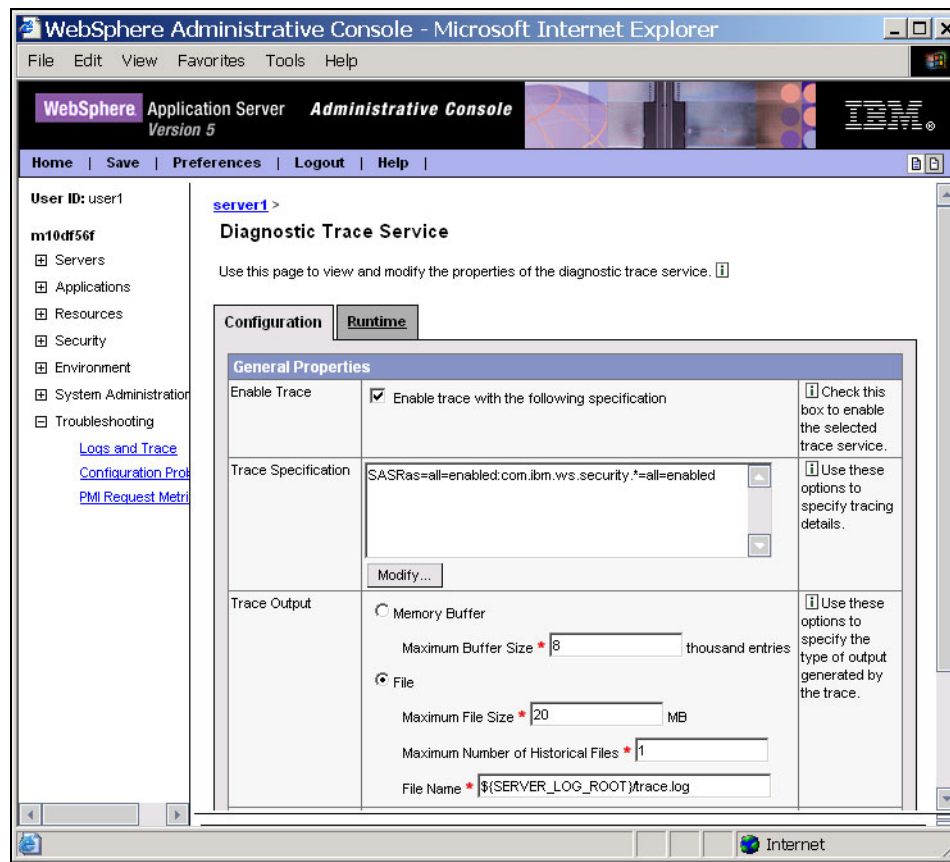


Figure 4-1 Administrative Console trace settings

Enabling trace at server startup

The diagnostic trace configuration settings for a server process determine the initial trace state for a server process. The configuration settings are read at server startup and used to configure the trace service. You can also change many of the trace service properties or settings while the server process is running.

The following process can be used to enable trace at server startup:

1. Start the Administrative Console.
2. Click **Troubleshooting** -> **Logs and Trace** in the console navigation tree, then click **Server** -> **Diagnostic Trace**.
3. Click **Configuration**.
4. Select the **Enable Trace** check box to enable trace, clear the check box to disable trace.
5. Set the trace specification to the desired state by entering the proper TraceString or click **Modify** to use the GUI interface.
6. Choose whether to direct trace output to either a file or an in-memory circular buffer.
7. (Optional) If the in-memory circular buffer is selected for the trace output, set the size of the buffer, specified in thousands of entries. This is the maximum number of entries that will be retained in the buffer at any given time.
8. (Optional) If a file is selected for trace output, set the maximum size, in megabytes, to which the file should be allowed to grow. When the file reaches this size, the existing file will be closed, renamed, and a new file with the original name reopened. The new name of the file will be based upon the original name with a timestamp qualifier added to the name. In addition, specify the number of history files to keep.
9. Select the desired format for the generated trace.
10. Save the changed configuration.
11. Start the server.

Enabling trace on a running server

You can modify the trace service state that determines which components are being actively traced for a running server by using the following procedure.

1. Start the Administrative Console.
2. Click **Troubleshooting** -> **Logging and Tracing** in the console navigation tree, then click **server** -> **Diagnostic Trace**.
3. Select the **Runtime** tab.
4. (Optional) Select the **Save Trace** check box if you want to write your changes back to the server configuration.
5. Change the existing trace state by changing the trace specification to the desired state.
6. (Optional) Configure the trace output if a change from the existing one is desired.

7. Click **Apply**.

Enabling trace on an application client or stand-alone process

Many stand-alone processes and the WebSphere J2EE application client define a process-specific mechanism for enabling trace. To enable trace and message logging for application clients or processes that have not defined such a mechanism, add the `-DtraceSettingsFile=filename` system property to the startup script or command.

For example, to trace the stand-alone client application program named `com.ibm.sample.MyClientProgram`, enter the command:

```
java -DtraceSettingsFile=MyTraceSettings.properties com.ibm.samples.  
MyClientProgram
```

The file, identified by file name, must be a properties file placed in the classpath of the application client or stand-alone process. An example file is provided `<WebSphere_root>/properties/TraceSettings.properties`.

To enable application client tracing, the following procedure can be used:

1. (Optional) Configure the `MyTraceSettings.properties` file to send trace output to a file. The `traceFileName` property in the trace settings file specifies one of two options:
 - The fully qualified name of an output file, for example, `traceFileName=\tmp\MyTraceFile.log`. You must specify this property to generate visible output.
 - `stdout`; when specified, the output is written to `System.out`.
2. (Optional) Specify a trace string for writing messages. The `Trace String` property specifies a startup trace specification, similar to that available on the server.

Tip: The key piece of information regarding WebSphere Application Server tracing can be found by clicking:

Troubleshooting -> Logs and Trace -> servername -> Diagnostic trace

Taking JMS traces within WebSphere

1. From the WebSphere Administrator's Console, select: **Default server -> Service Tab -> Trace services**
2. Then set `com.ibm.ejs.jms.*=all=enabled`

Note: The previous setting provides WebSphere JMS Wrapper trace only. If you need full JMS trace, use the programmatic method shown above.

4.3.8 First Failure Data Capture tool

When there is a processing failure, the First Failure Data Capture tool saves the information in logs. The First Failure Data Capture tool is primarily used by IBM Service to help diagnose problems. An IBM service representative may request these logs to investigate a problem. The First Failure Data Capture logs are located in <WebSphere_root>/logs/ffdc.

You should use the other logs and tools described for problem determination first since they contain error messages that will help you find the problem.

4.3.9 Other WebSphere Application Server logs

There is an assortment of other WebSphere Application Server logs that may be helpful after you review the JVM and service logs. These logs are all text-based and can be opened with your favorite text viewer.

startServer.log and stopServer.log

These logs contain start and stop information for the server process. If you use the `-trace` option on the `startServer` command, that trace information goes in these files. The JVM logs contain more information about the startup process. These logs are in the server's log folder.

addNode.log and removeNode.log

The `addNode.log` and `removeNode.log` contain messages and errors stemming from the process of adding and removing nodes. If your node fails to add or remove from a cell, review these logs for more information. If you use `-trace` on the `addNode` or `removeNode` commands, the trace information will be in these logs.

wsadmin.traceout

The `wsadmin.traceout` file is the default file for the tracing and logging information from `wsadmin`. It is located in <WebSphere_root>/logs.

serverStatus.log

This log contains the server status. This log is in the server's log folder.

http_plugin.log

This log records the plug-in loading and system information. This is in the <WebSphere_root>/logs folder. Use this log when troubleshooting plug-in problems.

<server_name>.pid

This file is created for a running server to store the PID number. You can use this file if you need to check if the process is running or if you need to kill it. It is located in the server's log folder. If you do not see this file, your server is not running.

PMEinstallSummary.log

If you need to review the WebSphere Application Server Enterprise features that were installed, you can see them in this log.

4.3.10 BackupConfig and RestoreConfig

BackupConfig is not a problem determination tool, but it is important to back up your WebSphere Application Server. Keeping backups will allow you to return to a working configuration or to have something with which to compare your current configuration. BackupConfig will create a snapshot file with your config files and application information. These commands are located in the WebSphere Application Server bin directory.

4.3.11 DumpNameSpace

The **dumpNameSpace** command lists the name space for you to review for the presence of your JNDI names. To run **dumpNameSpace**, go the WebSphere Application Server bin directory.

```
./dumpNameSpace.sh -host myhost.mycompany.com -port 2809
```

Use the **-help** option for a list of **dumpNameSpace** options.

Since the **dumpNameSpace** result can be long, you may want to pipe or redirect the output.

4.3.12 Tivoli® Performance Viewer

Tivoli Performance Viewer allows you to monitor your WebSphere Application Server application servers. Although you would normally use the tool for analyzing performance, you may find it useful for problem determination related to poor performance or eventual system failure. For example, you could monitor

database connections and discover that part of your code must not be closing connections if they are used and never freed.

Tivoli Performance Viewer can be installed during the WebSphere Application Server install. To enable and start the Tivoli Performance Viewer, follow these steps.

1. In the Administrative Console, go to **Servers -> Application Servers -> <your_server>**.
2. Click **Performance Monitoring Service**.
3. Check the **Startup** box and select a monitoring level. You can select the same level for all or use the custom setting and set each component separately.
4. If your server is part of a cell, set the Performance Monitoring Service on the nodeagent as well.
5. Restart the server and the nodeagent.
6. Using a machine with Tivoli Performance Viewer installed, go to the WebSphere Application Server bin directory and run the **tperfviewer** command, using the hostname of the system to monitor the RMI or SOAP port. You will need to supply a username and password if security is enabled.
UNIX with RMI: `./tperfviewer.sh <hostname> 9809 RMI`
UNIX with SOAP: `./tperfviewer.sh <hostname> 8879 SOAP`
7. Use the tool to monitor system resource usage. Tivoli Performance Viewer will suggest changes to improve performance based on the data gathered from your system.

4.4 WebSphere MQ

Version: WebSphere MQ V5.3.0.4

WebSphere MQ, for most installations, is run as a client/server application which incorporates many different platforms and operating systems that interact with each other. You will often need to review data from many of the components that make up the WebSphere MQ environment. These could include other z/OS systems, UNIX systems from many other vendors, or Windows.

The most important diagnostic data to review is written to the MQ error log and the associated FDC files. These diagnostic files may be created on both the Server (where the QueueManager resides) and the Client, so it is important to review both Client and Server for the relevant files.

4.4.1 First-Failure Support Technology (FFST™)

The Function Stack and Trace History of FFST are used by IBM to assist in problem determination. In most cases, there is little that the system administrator can do when an FFST record is generated, apart from discussing problems with the IBM Support Center.

However, there are some problems that the system administrator might be able to solve. If the FFST shows “out of resource” or “out of space” on device descriptions when calling one of the IPC functions (for example, `semop` or `shmget`), it is likely that the relevant kernel parameter limit has been exceeded.

If the FFST report shows a problem with `setitimer`, it is likely that a change to the kernel timer parameters is needed.

FFST: WebSphere MQ for UNIX systems

For WebSphere MQ for UNIX systems, FFST information is recorded in a file in the `/var/mqm/errors` directory.

These errors are normally severe, unrecoverable errors, and indicate either a configuration problem with the system or a WebSphere MQ internal error.

The files are named `AMQnnnnn.mm.FDC`, where `nnnnn` is the ID of the process reporting the error and `mm` is a sequence number, normally 0.

When a process creates an FFST record, it also sends a record to `syslog`. The record contains the name of the FFST file assist in automatic problem tracking.

The `syslog` entry is made at the `user.error` level. See the operating system documentation about `syslog.conf` for information about configuring this.

Example 4-7 Sample FFST output

```
+-----+
| WebSphere MQ First Failure Symptom Report
| =====
|
| Date/Time :- Friday March 15 17:56:51 SGT 2002
| Host Name :- sunrts3 (SunOS 5.7)
| PIDS :- 5724B4102
| LVLS :- 530
| Product Long Name :- WebSphere MQ for Sun Solaris
| Vendor :- IBM
| Probe Id :- RM161000
| Application Name :- MQM
| Component :- rrmChangeClq
| Build Date :- Mar 13 2002
|
```

```
| CMVC level :- p000-L020312
| Build Type :- IKAP - (Production)
| UserID :- 00001001 (mqm)
| Program Name :- amqrrmfa
| Process :- 00019454
| Thread :- 00000001
| QueueManager :- REGR
| Major Errorcode :- rrcE_CLUS_COMMAND_ERROR
| Minor Errorcode :- OK
| Probe Type :- MSGAMQ9413
| Probe Severity :- 2
| Probe Description :- AMQ9413: Repository command format error, command code
| 0
| FDCSequenceNumber :- 0
```

```
+-----+
MQM Function Stack
rrmProcessMsg
rrmChangeClq
xcsFFST
MQM Trace History
---{ zstVerifyPCD
---} zstVerifyPCD rc=OK
---{ ziiMQCMIT
----{ ziiCreateIPCCMessage
-----{ zcpCreateMessage
-----} zcpCreateMessage rc=OK
----} ziiCreateIPCCMessage rc=OK
----{ ziiSendReceiveAgent
-----{ zcpSendOnPipe
...

```

Tip: Refer to the following documents if you need further assistance:

- ▶ *MQ System Administration Guide*
- ▶ *WebSphere MQ for AIX, V5.3 Quick Beginnings*
- ▶ *WebSphere MQ for HP-UX, V5.3 Quick Beginnings*
- ▶ *WebSphere MQ for Linux for Intel and Linux for zSeries, V5.3 Quick Beginnings*
- ▶ *WebSphere MQ for Solaris, V5.3 Quick Beginnings*

4.4.2 Error logs

WebSphere MQ uses a number of error logs to capture messages concerning the operation of WebSphere MQ itself, any queue managers that you start, and

error data coming from the channels that are in use. The location of the error logs depends on whether the queue manager name is known and whether the error is associated with a client.

4.4.3 WebSphere MQ error logs

WebSphere MQ uses a number of error logs to capture messages concerning the operation of WebSphere MQ itself, any queue managers that you start, and error data coming from the channels that are in use.

The location of the error logs depends on whether the queue manager name is known and whether the error is associated with a client.

WebSphere MQ for UNIX systems

If the queue manager name is known and the queue manager is available, error logs are located in: `/var/mqm/qmgrs/<QM_NAME>/errors`.

If the queue manager is not available, error logs are located in: `/var/mqm/qmgrs/@SYSTEM/errors`.

If an error has occurred with a client application, error logs are located on the client's root drive in: `/var/mqm/errors`.

Log files

At installation time, an @SYSTEM errors subdirectory is created in the qmgrsfile path. The errors subdirectory can contain up to three error log files named AMQERR01.LOG, AMQERR02.LOG and AMQERR03.LOG.

After you have created a queue manager, three error log files are created when the queue manager needs them. These files have the same names as the @SYSTEM ones (AMQERR01, AMQERR02, and AMQERR03), and each has a capacity of 256 KB. The files are placed in the errors subdirectory of each queue manager that you create.

As error messages are generated, they are placed in AMQERR01. When AMQERR01 gets bigger than 256 KB, it is copied to AMQERR02. Before the copy, AMQERR02 is copied to AMQERR03.LOG. The previous contents of AMQERR03, if any, are discarded. The latest error messages are thus always placed in AMQERR01, the other files being used to maintain a history of error messages.

All messages relating to channels are also placed in the appropriate queue manager's errors files unless the name of their queue manager is unknown or the queue manager is unavailable. When the queue manager name is

unavailable or its name cannot be determined, channel-related messages are placed in the @SYSTEM errors subdirectory.

To examine the contents of any error log file, use your usual system editor.

Early errors

There are a number of special cases where the above error logs have not yet been established and an error occurs. WebSphere MQ attempts to record any such errors in an error log. The location of the log depends on how much of a queue manager has been established.

If, due to a corrupt configuration file for example, no location information can be determined, errors are logged to an errors directory that is created at installation time on the root directory (/var/mqm or C:\MQM).

If the WebSphere MQ configuration file is readable, and the DefaultPrefix attribute of the AllQueueManagers stanza is readable, errors are logged in the errors subdirectory of the directory identified by the DefaultPrefix attribute.

Operator messages

Operator messages identify normal errors, typically caused directly by users doing things like using invalid command parameters. Operator messages are national language enabled, with message catalogs installed in standard locations.

The messages, if any, are written to the associated window. In addition, some operator messages are written to the AMQERR01.LOG file in the queue manager directory, and others to the @SYSTEM directory copy of the error log.

Dead-letter queues

Messages that cannot be delivered for some reason are placed on the dead-letter queue. You can check whether the queue contains any messages by issuing an **MQSC DISPLAY QUEUE** command. If the queue contains messages, you can use the provided browse sample application (amqsbcbg) to browse messages on the queue using the **MQGET** call. The sample application steps through all the messages on a named queue for a named queue manager, displaying both the message descriptor and the message context fields for all the messages on the named queue. You must decide how to dispose of any messages found on the dead-letter queue, depending on the reasons for the messages being put on the queue. Problems may occur if you do not associate a dead-letter queue with each queue manager.

If a channel ceases to run for any reason, applications will probably continue to place messages on transmission queues, creating a potential overflow situation. When this occurs in a message-originating node, and the local transmission

queue is full, the application's PUT fails. When this occurs in a staging or destination node, there are different ways in which the MCA copes with the situation:

- ▶ By calling the message-retry exit, if one is defined.
- ▶ By directing all overflow messages to a dead-letter queue (DLQ), returning an exception report to applications that requested these reports.

Note: In distributed queuing management, if the message is too big for the DLQ, the DLQ is full, or the DLQ is not available, the channel stops and the message remains on the transmission queue. Ensure your DLQ is defined, available, and sized for the largest messages you will handle.

- ▶ By closing down the channel, if neither of the previous options succeeded.
- ▶ By returning the undelivered messages back to the sending end and returning a full report to the reply-to queue (MQRC_EXCEPTION_WITH_FULL_DATA and MQRO_DISCARD_MSG).

If an MCA is unable to put a message on the DLQ then check for the following:

- ▶ The channel stops.
- ▶ Appropriate error messages are issued at the system consoles at both ends of the message channel.
- ▶ The unit of work is backed out, and the messages reappear on the transmission queue at the sending channel end of the channel.
- ▶ Triggering is disabled for the transmission queue.

Configuration files

Configuration file errors typically prevent queue managers from being found, and result in queue manager unavailable type of errors. Ensure that the configuration files exist and that the WebSphere MQ configuration file references the correct queue manager and log directories.

4.4.4 Tracing WebSphere MQ for AIX

WebSphere MQ for AIX uses the standard AIX system trace. Tracing is a two-step process:

1. Gathering the data.
2. Formatting the results.

WebSphere MQ uses two trace hook identifiers:

▶ X'30D'

This event is recorded by WebSphere MQ on entry to or exit from a subroutine.

▶ X'30E'

This event is recorded by WebSphere MQ to trace data such as that being sent or received across a communications network.

Trace provides detailed execution tracing to help you to analyze problems.

IBM service support personnel may ask for a problem to be recreated with trace enabled. The files produced by trace can be very large so it is important to qualify a trace, where possible. For example, you can optionally qualify a trace by time and by component.

There are two ways to run trace:

- ▶ Interactively - the following sequence of commands runs an interactive trace on the program *myprog* and ends the trace.

```
trace -j30D,30E -o trace.file  
->!myprog  
->q
```

- ▶ Asynchronously - the following sequence of commands runs an asynchronous trace on the program *myprog* and ends the trace.

```
trace -a -j30D,30E -o trace.file  
myprog  
trcstop
```

You can format the trace file with the command:

```
trcrpt -t usr/mqm/lib/amqtrc.fmt trace.file > report.file
```

where *report.file* is the name of the file where you want to put the formatted trace output.

Note: All WebSphere MQ activity on the machine is traced while the trace is active.

4.4.5 WebSphere MQ Java tracing

WebSphere MQ JMS applications normally invoke trace by using command line arguments to the `java` command, for example:

```
java -DMQJMS_TRACE_LEVEL=base -DMQJMS_TRACE_DIR=\tmp\myfile.txt
myJMSApplication
```

Sometimes, it is not possible to set the trace level from the command line, for example when the JMS application is started as a servlet within an application server. In this case, it may be appropriate to invoke trace from within the source code.

This can be done as follows:

```
com.ibm.mq.jms.services.ConfigEnvironment.start();
//uses MQJMS_TRACE_LEVEL and _DIR from the system properties
com.ibm.mq.jms.services.ConfigEnvironment.start(String level);
//uses MQJMS_TRACE_DIR; the level parameter should "on", or "base"
com.ibm.mq.jms.services.ConfigEnvironment.stop();
```

The value of `MQJMS_TRACE_DIR` can be set programmatically with the following code:

```
java.util.Properties.props = System.getProperties();
props.put("MQJMS_TRACE_DIR", "my/directory/name");
props.put("DMQJMS_TRACE_LEVEL", "base");
System.setProperties(props);
```

Note: The file name is fixed to `mqjms.trc`; only the directory can be modified in this manner.

4.4.6 Formatting the MQ trace file

You can format the trace file with the command:

```
trcrpt -t mqmtop/lib/amqtrc.fmt trace.file > report.file
```

where `report.file` is the name of the file where you want to put the formatted trace output.

Note: All WebSphere MQ activity on the machine is traced while the trace is active.

Example 4-8 on page 74 shows a sample of the data contained in WebSphere MQ for an AIX trace.

Example 4-8 WebSphere MQ trace

```
30E 0.250173285 0.915095 MQS Thread stack
30E 0.250661751 0.488466 pid(10004)
30E 0.251111529 0.449778 MQS -> zlaMainThread
30E 0.251151575 0.040046 MQS -> zlaProcessMessage
30E 0.251152096 0.000521 MQS -> zlaProcessMQIRequest
30E 0.251152532 0.000436 MQS -> zlaMQGET
30E 0.251153015 0.000483 MQS -> zsqMQGET
30E 0.251191184 0.038169 MQS -> kpiMQGET
30E 0.251253521 0.062337 MQS -> kqiWaitForMessage
30E 0.251254039 0.000518 MQS -> kqiWaitForABit
30E 0.251291476 0.037437 MQS -> apiLockExclusive
30E 0.251940443 0.648967 MQS Returning an error to the AI Layer:
CompCode 2 Reason 7f1 MQRC_NO_MSG_AVAILABLE
30E 0.252257186 0.316743 hev=1::0::0-275688
30E 0.252258418 0.001232 hev=1::0::0-275604
30E 0.252263620 0.005202 hev=1::0::0-275688 Timeout(-1)
30E 0.252353122 0.089502 MQS Thread stack
30E 0.252366827 0.013705 pid(10004)
30E 0.253368728 1.001901 MQS -> ccxTcpResponder
30E 0.253369247 0.000519 MQS -> ccxResponder
30E 0.253369732 0.000485 MQS -> rrxResponder
30E 0.253370165 0.000433 MQS -> rriMQIServer
30E 0.253370603 0.000438 MQS -> MQGET
30E 0.253371040 0.000437 MQS -> zstMQGET
30E 0.253371499 0.000459 MQS -> ziiMQGET
30E 0.253371949 0.000450 MQS -> zcpDeleteMessage
```

Hint: A good starting point is to search for the string MQRC which identifies MQ reason codes. While many of these do not indicate a serious error, as in the above case, the key to your problem might be as simple as finding the reason code. The reason code is displayed in HEX, for example, 7f1. The messages are documented in the manual in HEX, X'07F1' and decimal, 2033.

Trace instructions

Unlike the other WebSphere MQ distributed platforms that use WebSphere MQ supplied internal tracing utilities, WebSphere MQ on the AIX platform uses AIX tracing to capture the WebSphere MQ trace data.

Issue **ps -ef** to display parent processes. This will assist with correlating the active MQ processes with the trace data since all processes will be traced.

There are two types of WebSphere MQ tracing to run on AIX: interactive and asynchronous.

Interactive trace

The following sequence of commands runs an interactive trace on the program myprog and ends the trace.

```
trace -j30D,30E -o trace_file
->!myprog
->q
```

Asynchronous trace

The following sequence of commands runs an asynchronous trace on the program myprog and ends the trace.

```
trace -a -j30D,30E -o trace_file
myprog
trcstop
```

Tracing

The output files in UNIX are always created in the directory /var/mqm/trace, with the file name AMQxxxxx (where xxxxx stands for an identification number).

There is one file for each active WebSphere MQ process.

4.4.7 WebSphere MQ Explorer

WebSphere MQ Explorer is available on Windows as a GUI interface to create, modify, and monitor your WebSphere MQ information. You can view remote queues and clusters on other WebSphere MQ systems. This is especially useful for working with WebSphere MQ installs on AIX and other operating systems.

In the eMerge scenario, we view our AIX queues from WebSphere MQ Explorer. To connect to a remote queue manager and cluster repository, there are several steps to follow. In our configuration, our listener port is set to the default 1414.

Using the WebSphere MQ Explorer, we can look for WebSphere MQ message related problems by reviewing the messages put on the queue, the queue depth, the monitor channels and other items.

Configuring remote queue managers in WebSphere MQ Explorer

The following steps describe the configuration of remote queue managers.

1. On the remote machine's queue manager, you need to add a server connection channel called SYSTEM.ADMIN.SVRCONN. To define the channel, run this command on your queue manager: **DEFINE CHANNEL(SYSTEM.ADMIN.SVRCONN) CHLTYPE(SVRCONN)**.

2. Create a listener on your remote queue manager. On Windows, a listener is set up by default on port 1414.

To create a TCP/IP listener on a UNIX system you need to edit the `/etc/services` and the `inetd.conf` files.

- a. Open the `/etc/services` file as a superuser or root and add:

```
MQSeries 1414/tcp
```

- b. Open the `inetd.conf` file and add this to call the `amqcrsta` program, using your queue manager name and WebSphere MQ install home.

```
MQSeries stream tcp nowait mqm <WebSphereMQ_root>/bin/amqcrsta amqcrsta  
-m <your_queue_manager?
```

- c. Restart the listener:

```
refresh -s inetd
```

3. Open WebSphere MQ Explorer on Windows. You should find it under **Start -> Programs -> IBM WebSphere MQ -> WebSphere MQ Explorer**.
4. Add a remote queue manager or a remote cluster.

– Remote queue manager:

- i. Expand WebSphere MQ, right-click **Queue Managers** and select **Show Queue Manager**.
- ii. Select **Show a remote queue manager**.
- iii. Enter your queue manager name and the connection name. The connection name is usually the hostname or IP address of the queue manager's system followed by the port in parentheses.
- iv. Click **OK**.

– Remote cluster:

- i. Expand WebSphere MQ, right-click **Clusters** and select **Show Cluster**.
- ii. Enter the cluster name, for example: `mycluster`.
- iii. Select **Repository queue manager is remote**.
- iv. Enter the queue manager name of your remote repository and a connection name. The connection name is usually the hostname or IP address of the repository queue manager's system. To add a port number, use parentheses and the port number appended to the hostname, for example: `mymachine(1414)`.
- v. Click **OK**.

If you receive an authorization error when attempting to connect, add a matching user to either the remote or local system. On the remote side, the user needs to be part of the mqm user group.

Using WebSphere MQ Explorer to review and monitor

Once you can view either local or remote queues and clusters, you can review information about them to perform problem determination.

Refresh

It is important to refresh your view frequently, otherwise you may not see messages being added or removed or other changes. To refresh all the queues for a manager, highlight the **Queues** folder on the left and then refresh. To refresh a single queue, highlight the queue and refresh. The same process follows for other views with channels, connections, etc.

Choose columns

To rearrange the information that is displayed, select **View -> Choose Columns...** then select and rearrange the columns that you would like to see first. For instance, moving Current Depth higher on the list makes it easy to review how many messages are on the queue. You can also resort the list by column.

Working with a single queue

If you right-click a message queue, there are a variety of options to work with:

1. Browse messages
2. Put a test message on the queue
3. View the status of the queue
4. Clear the queue of the current messages
5. Review the properties of the queue

You can put test messages on the queue to check that they are received. You may want to clear your messages in order to start a test again.

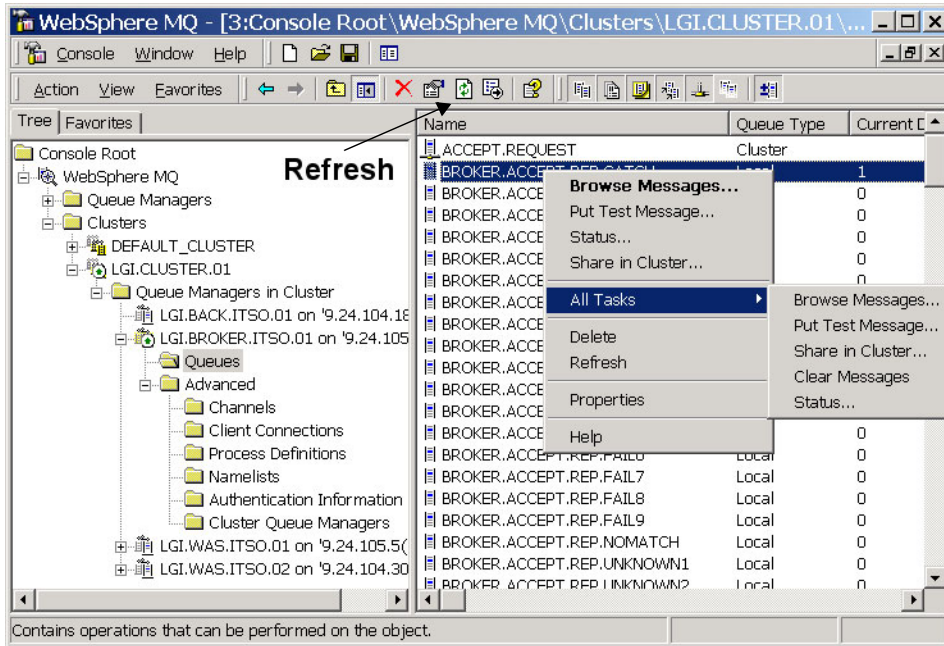


Figure 4-2 WebSphere MQ Explorer with queue menu

You can review messages that are waiting on the queue by selecting **Browse Messages**. To inspect a single message, select a message and click **Properties**.

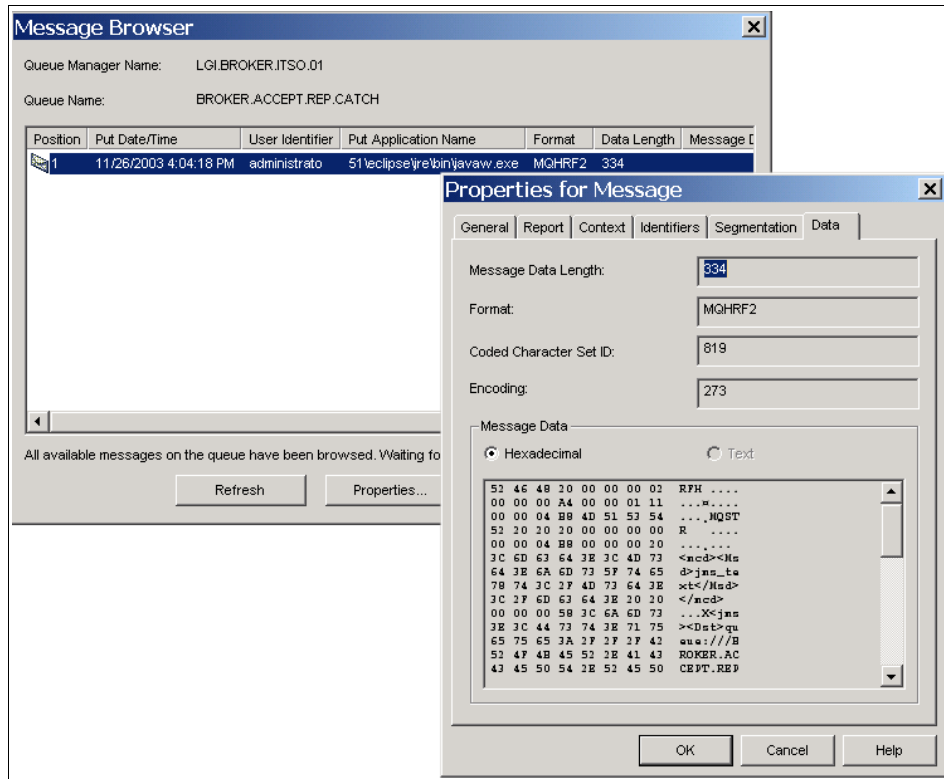


Figure 4-3 WebSphere MQ Explorer viewing the data in a message

Channels

You can check that your channels are running and start or stop them if necessary. If you see problems with your messages, you should check to see if the channels are still running. To review channels, you can expand a queue manager and its Advanced folder. Click **Channels**, right-click a channel and select **Status**. You can also rearrange your columns to make the Overall Channel Status easily viewable.

Queue manager

To see what applications, users or channels have a queue open, right-click a queue and select **Status**.

4.5 WebSphere Business Integration Message Broker

Version: WebSphere Business Integrator Message Broker V5

The following information is required to assist with WebSphere Business Integration Message Broker (referred to as Message Broker later in this chapter) problem diagnosis on distributed platforms.

- ▶ The failing messageflow
- ▶ The input message to drive messageflow (and messageset if using MRM)
- ▶ Database definition of user tables (DDL)
- ▶ Clear description of the data being inserted

Recreate the error condition and provide:

1. Message Broker service trace.
2. A service level trace of the broker admin agent during a failed deploy that causes the BIPxxxxE (where xxxx stands for an identifier number) message.
3. A service level trace of the brokers “default” execution group that is experiencing the problem.
4. The current environment.
5. The abend file that is generated during tracing.

These two traces should be obtained from the time the broker and execution have been started, and then extracted after the failure has occurred. This can be done by using the following procedure.

4.5.1 Tracing

You may decide that you want to take a closer look at what is going on inside WebSphere Business Integration Message Broker but wish to use an alternative method to the Debugger. This alternative is tracing. Tracing provides more details about what is happening while code executes. The tracing types focused on here are the user trace, service trace and ODBC trace. User tracing is typically used for debugging applications and can trace execution groups or individual flows. Service tracing allows more comprehensive tracing. ODBC tracing allows tracing of the interaction between WebSphere Business Integration Message Broker and the database.

As a general rule of thumb, you should begin with WebSphere Business Integration Message Broker’s user trace, and move on to the ODBC or service trace if the user trace points you in that direction. After the ODBC trace, if you still

need more information, investigate the db2diag.log (see “db2diag.log file” on page 95).

Tracing the message broker

Follow the steps below to perform tracing for the message broker.

1. Run `mqsistop <broker_name>`
2. Run `print -n debug > /var/mqsi/registry/<broker_name>/Trace` where `<broker_name>` is the name of your broker.
3. Run `mqsistart <broker_name>`
4. Recreate the problem.
5. Make sure the broker is stopped.
6. Run `mqsireadlog brokerName -t -b agent -f -o agent.xml`
7. Run `mqsireadlog brokerName -t -b service -f -o service.xml`
8. Run `mqsiformatlog -i agent.xml -o agent.fmt`
9. Run `mqsiformatlog -i service.xml -o service.fmt`
10. The trace file you generated from the earlier `ksh` `print` command can now be deleted.

Tracing the Message Broker execution group

The following steps describe how to trace the message broker execution group.

1. Start with `mqsistop <broker_name>`
2. Run

```
print -n "debug" > /var/mqsi/registry/<broker_name>/executionGroupTraceOverrideLevel
```

where `<broker_name>` is the name of your broker and `print` is a `ksh` command.
3. Run

```
mqsistart <broker_name>
```

or you can dynamically start the user trace by issuing the following command:

```
mqsichangetrace <broker_name> -u -e executionGroup debug -r
```

Wait until the `DataFlowEngine` process has failed; this should generate trace files in the log directory.
4. Run:

```
mqsireadlog brokerName -t -e <exeGrpLabel> -f -o eg.xml
```

where `<exeGrpLabel>` is the execution group label `Eg` default.

5. Run:

```
mqsiformatlog -i eg.xml -o eg.fmt
```

6. The trace file you generated from the earlier **ksh** print command can now be deleted.
7. Send the agent.fmt, service.fmt and the eg.fmt files to the support.

The request customer clears the service trace files (using **mqsichangetrace -r**) and ensures that it does not wrap by assigning a large enough size (using the **-c** option). Additionally, you should clear the ODBC trace file prior to running the failure test and, if possible, recycle the broker (using **mqsistop/mqsistart**) since the ODBC trace will then show the database connection being re-established.

User trace

User trace causes additional processing power to be used and generates large amounts of data. This will have an impact on performance. For this reason and to make the trace easier to understand, you should try to restrict the trace to be as narrow as possible and only active for as short a period as necessary. It is inactive by default and must be started explicitly. Brief instructions, which should be sufficient for most cases, are given. For more information, see *WebSphere Business Integration Message Broker V5.0 Diagnosis* at:

ftp://ftp.software.ibm.com/software/integration/wbibrokers/docs/messagebroker_Diagnosis.pdf

You can start trace at three levels:

- ▶ **normal**: This tracks events that affect objects you create and delete, such as message flow nodes.
- ▶ **debug**: This tracks the beginning and the end of processes, and monitors objects that are affected by that process.
- ▶ **none**: Tracing is inactive.

To start tracing of an execution group, issue the following command:

```
mqsichangetrace <BROKER_NAME> -u -e <GROUP> -l LEVEL
```

where *BROKER_NAME* is the name of your broker, *GROUP* is the name of the execution group and *LEVEL* is the level of trace (normal or debug).

To start tracing of an individual message flow, issue the following command:

```
mqsichangetrace <BROKER_NAME> -u -e <GROUP> -f FLOW -r -l LEVEL
```

where *BROKER_NAME* is the name of your broker, *GROUP* is the name of the execution group, *LEVEL* is the level of trace (normal or debug) and *FLOW* is the name of the flow to be debugged.

You can now start tracing both execution groups and message flows from the workbench by following these instructions:

1. Switch to the Broker Administration perspective.
2. In the Domains view, right-click the message flow or execution group that you wish to trace.
3. Select **User Trace** and then the level of tracing.

A dialog should appear explaining that a successful response was received from the Configuration Manager, and an alert should be displayed in the Alerts view stating that the tracing is enabled. If the alert is not displayed, check that the WebSphere MQ channels are running.

Note: To switch trace off, set the trace level to none.

To see the current tracing status, issue the following command:

```
mqsiporttrace <BROKER_NAME> -u -e <GROUP> [-f FLOW]
```

where *BROKER_NAME* is the name of your broker, *GROUP* is the name of the execution group, and *FLOW* is the name of the flow status which is to be reported upon. Note that the flow is optional and that the report on an execution group will return none, even if one or more flows are set to normal or debug. An example output is shown in Example 4-9.

Example 4-9 mqsiporttrace command example

```
$ mqsiporttrace -u LGI_BROKER_ITSO_01 -e default -f ReplyCustDetails  
BIP8098I: Trace level: normal, mode: safe, size: 4096 KB.
```

```
BIP8071I: Successful command completion.
```

To retrieve the trace data recorded by the user trace, use the **mqsi readlog** command. This will write the trace to a file or the command prompt in XML format. Use a command like:

```
mqsi readlog <BROKER_NAME> -u -e <GROUP> -o trace.xml
```

where *BROKER_NAME* is the name of your broker, *GROUP* is the name of the execution group, and the file to output the trace to is trace.xml. Since this trace is fairly unreadable, it is almost certain that you will want to format it before attempting to read it.

To format the retrieved trace, issue the following command:

```
mqsi formatlog -i trace.xml -o formattrace.log
```

where trace.xml is the trace to be formatted and formattrace.log is the file where the formatted trace should be written.

An example of taking, reading and formatting a trace is given in Example 4-10. The output it generates is shown in example Example 4-11.

Example 4-10 The commands to trace message flow RequestCustDetails

```
$ mqsichangetrace LGI_BROKER_ITS0_01 -u -e default -r
BIP8071I: Successful command completion.
$ mqsichangetrace LGI_BROKER_ITS0_01 -u -e default -f RequestCustDetails -l normal
BIP8071I: Successful command completion.
$ mqsichangetrace LGI_BROKER_ITS0_01 -u -e default -f RequestCustDetails -l none
BIP8071I: Successful command completion.
$ mqsireadlog LGI_BROKER_ITS0_01 -u -e default -o trace.xml
BIP8071I: Successful command completion.
$ mqsiformatlog -i trace.xml -o formattrace.log
BIP8071I: Successful command completion.
```

It is a good idea to write batch files that run these commands to start and stop trace. A formatted trace output for a successful inquiry of customer details from the eMerge application is created using the commands shown above.

Example 4-11 Trace output (due to page restrictions, each line has been split)

Timestamps are formatted in local time, 300 minutes before GMT.

```
2003-12-08 17:48:43.641231      6169  UserTrace  BIP2632I: Message received and propagated to
'out' terminal of MQ input node 'RequestCustDetails.CUST.DETAILS.REQUEST'.
2003-12-08 17:48:43.641456      6169  UserTrace  BIP6060I: Parser type 'Properties' created on
behalf of node 'RequestCustDetails.CUST.DETAILS.REQUEST' to handle portion of incoming message
of length 0 bytes beginning at offset '0'.
2003-12-08 17:48:43.641674      6169  UserTrace  BIP6061I: Parser type 'MQMD' created on
behalf of node 'RequestCustDetails.CUST.DETAILS.REQUEST' to handle portion of incoming message
of length '364' bytes beginning at offset '0'. Parser type selected based on value 'MQHMD' from
previous parser.
2003-12-08 17:48:43.641746      6169  UserTrace  BIP6061I: Parser type 'MQRFH2' created on
behalf of node 'RequestCustDetails.CUST.DETAILS.REQUEST' to handle portion of incoming message
of length '212' bytes beginning at offset '364'. Parser type selected based on value 'MQRHF2'
from previous parser.
2003-12-08 17:48:43.642120      6169  UserTrace  BIP6061I: Parser type 'XML' created on behalf
of node 'RequestCustDetails.CUST.DETAILS.REQUEST' to handle portion of incoming message of
length '77' bytes beginning at offset '576'. Parser type selected based on value 'XML' from
previous parser.
2003-12-08 17:48:43.643207      6169  UserTrace  BIP4124I: Message propagated to 'out'
terminal of compute node 'RequestCustDetails.Check all fields present'.
2003-12-08 17:48:43.643764      6169  UserTrace  BIP4004I: Message propagated to 'true'
terminal of filter node 'RequestCustDetails.Field.check + version'.
```

```
2003-12-08 17:48:43.645267      6169  UserTrace  BIP4124I: Message propagated to 'out'
terminal of compute node 'RequestCustDetails.Transform Request'.
2003-12-08 17:48:43.645423      6169  UserTrace  BIP4005I: Message propagated to 'false'
terminal of filter node 'RequestCustDetails.Check Request Target'.
2003-12-08 17:48:43.646339      6169  UserTrace  BIP4201I: Message propagated to out terminal
from node 'RequestCustDetails.Set Domain to jms_text'.
      A reset content descriptor node has received a message
and is propagating it to any nodes connected to its out terminal.
      No user action required.
2003-12-08 17:48:43.647411      6169  UserTrace  BIP2638I: The MQ output node
'RequestCustDetails.Forward Request' attempted to write a message to queue
'INQUIRECUST.REQUEST' connected to queue manager ''. The MQCC was '0' and the MQRC was '0'.
2003-12-08 17:48:43.647499      6169  UserTrace  BIP2622I: Message successfully output by
output node 'RequestCustDetails.Forward Request' to queue 'INQUIRECUST.REQUEST' on queue
manager ''.
```

Threads encountered in this trace:
6169

Service trace

Service trace is the second of the two main types of trace offered by WebSphere Business Integration Message Broker. It allows more comprehensive tracing of the various components. You can also trace the execution of commands, including the trace commands. Service trace cannot be initiated from the workbench, you must use the command line.

Note: You are advised to activate service traces only when you receive an error message that instructs you to start service trace, or when directed to do so by your IBM Support Center.

Enabling of service trace is very similar to that of user trace (the `-u` is replaced with `-t`) and is detailed in the *WebSphere Business Integration Message Broker V5 Diagnosis* book. No more details are given here, since this level of trace will not be required in most circumstances.

ODBC trace

If you suspect that you may have database problems, you may be required to perform an ODBC trace. ODBC stands for Open Database Connectivity, an open standard application programming interface for accessing databases. It is how WebSphere Business Integration Message Broker accesses both the broker's own database and any user databases used by the message flows.

On UNIX, edit the `odbc.ini` file to initiate trace for ODBC activity (if you are using the default `odbc.ini` file, this is located at `/var/wmqi/odbc/.odbc.ini`):

1. Under the stanza entry [ODBC] change Trace=0 to be Trace=1; an example stanza entry is shown in Example 4-12.
2. Optionally, modify the TraceFile file path value to a preferred value. Note that all the trace records go to a single file, which can be confusing with multiple flows or execution groups.
3. Restart the broker. Tracing will not begin until the server is restarted.

Example 4-12 Sample stanza in .odbc.ini file with trace on

```
[ODBC]
Trace=1
TraceFile=/var/wmqi/odbc/odbctrace.out
TraceDll=/usr/opt/wmqi/merant/lib/odbctrac.so
InstallDir=/usr/opt/wmqi/merant
```

Note that as the server is restarted, the trace will also contain the broker's startup access to the database. There is a lot of data generated from this trace. For this reason, you may find it useful to use a command such as the AIX `tail` command to monitor the file as you perform the failing operation. An example entry into the trace file is shown in Example 4-13.

Example 4-13 Sample entry in UNIX odbc trace file

```
ppid=15582:pid= 5208:1b1c      ENTER SQLPrepare
HSTMT                0x3caf6558
UCHAR *              0x3ac90f38 [ -3] "DELETE FROM DB2INST1.BAGGREGATE WHERE replyGroupId = ?"
SDWORD               -3
```

The trace node

The trace node is used to generate trace information including text, message content and date and time information that can be used to monitor the behavior of the message flow.

The generated trace can be written to the user trace file, a separated defined file or the local error log.

Note: Trace node output is independent of user trace. A Trace node that is set up to generate trace will generate trace even when user trace is set to none.

The trace node is one of the many nodes available when developing message flows with the workbench. It can be used to debug an operating system, but is probably more suited to use during testing. A trace node has two connectors, in and out. Any existing connection can be replaced with a connection via a trace

node. The properties of the trace node are set by right-clicking the node and selecting **Properties**, see Example 4-4.

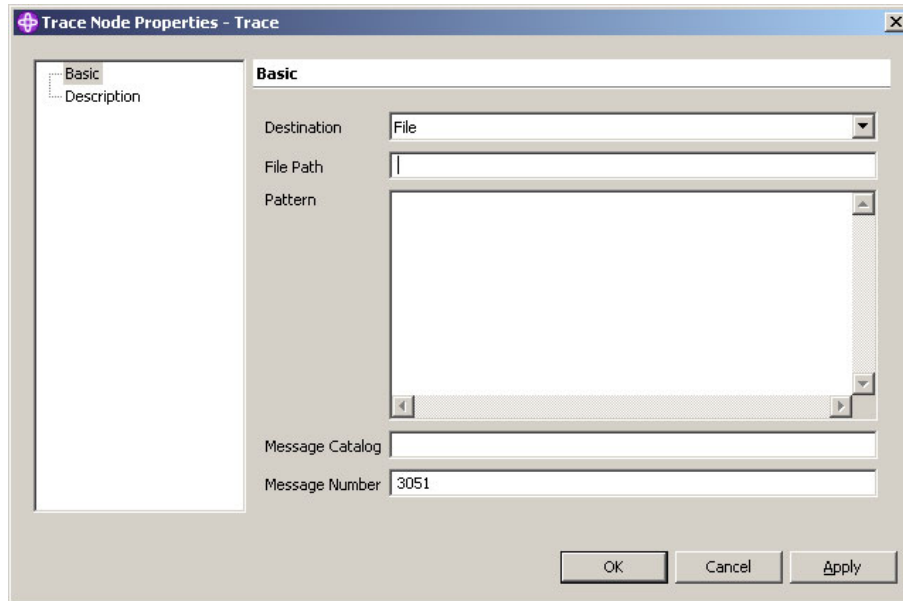


Figure 4-4 Trace Node Properties dialog box

The Destination field can be set to None, User Trace, File or Local Error Log. None will write output. User Trace will go into the user trace's files, regardless of the user trace settings. File will output to the specified file in File Path. Local Error Log will write out to the destination of usual log information, for instance the Event Viewer in Windows and syslog for UNIX. If you are writing to the local error log, then you must use a message catalog or create a user-defined one. Both this and the message number must be specified in the properties.

The pattern is what is to be outputted, specified in ESQL. This pattern:

```
Message Contents:
${Root}
Time is: ${EXTRACT(HOUR FROM CURRENT_TIMESTAMP)} :${EXTRACT(MINUTE FROM
CURRENT_TIMESTAMP)}
```

will produce output as seen in Example 4-14 on page 88.


```

(0x03000000):MsgFlags          = 0
(0x03000000):OriginalLength   = -1
)
(0x01000000):MQRFH2           = (
(0x03000000):Version           = 2
(0x03000000):Format            = 'MQSTR  '
(0x03000000):Encoding          = 273
(0x03000000):CodedCharSetId    = 1208
(0x03000000):Flags             = 0
(0x03000000):NameValueCCSID    = 1208
(0x01000000):mcd               = (
(0x01000000):Msd = (
(0x02000000): = 'jms_text'
)
)
(0x01000000):jms               = (
(0x01000000):Dst = (
(0x02000000): = 'queue:///BROKER.INQUIRECUST.REQUEST'
)
(0x01000000):Tms = (
(0x02000000): = '1071002810229'
)
(0x01000000):Dlv = (
(0x02000000): = '1'
)
)
)
(0x01000000):BLOB              = (
(0x03000000):BLOB              =
X'3c42524f4b45523e3c5156455253494f4e3e313c2f5156455253494f4e3e3c4355535449443e323c2f43555354494
43e3c2f42524f4b45523e'
(0x03000000):UnknownParserName = 'jms_text'
)
)
Time is: 15:47

```

If you try to deploy a flow containing a trace node which has syntax errors, the flow will not deploy. This is contrary to the information given in the *WebSphere Business Integration Message Broker V5 Message Flows* book. Instead, the syntax error should be given in detail in the Event Log of the workbench for the broker on which deployment was attempted.

Further details on how to use the trace node can be found in the *WebSphere Business Integration Message Broker V5 Message Flows* book.

4.5.2 Message Broker Debugger

The debugging facility is part of the Eclipse tooling with which flows are developed for WebSphere Business Integration Message Broker. This allows the user to examine the state of the message as it passes through the flow, adding and removing breakpoints dynamically. Hopefully, this will aid the user in the task of determining any problems.

Note: The Agent Controller must be installed and started on the WebSphere Business Integration Message Broker server on which the flows to be debugged are running.

1. To begin debugging, load the Eclipse-based toolkit from the start menu.
 - a. Click **Start -> Programs -> IBM WebSphere Business Integration Message Brokers -> Message Brokers Toolkit.**
2. Open the Flow Debug perspective.

Click **Window -> Open Perspective -> Flow Debug.**
3. Connect to the WebSphere Business Integration Message Broker server on which the flows to be debugged are running.
 - a. Right-click in the Flow Debug view and click **Attach.**
 - b. Type the server's hostname or IP address into the input field and click **Add.**
 - c. Click **Next.**
 - d. Select the required flow engine (there is probably only one) and click **Finish.**

The Flow Debug view now shows all the message flows running on the engine. An example of this can be seen in Figure 4-5.

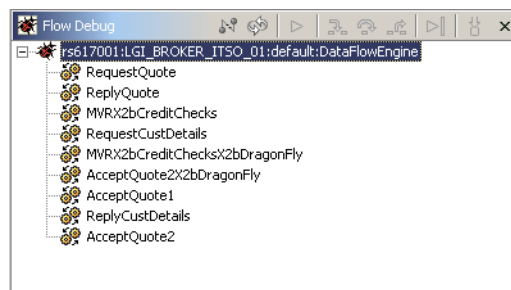


Figure 4-5 Example Flow Debug view

4. Open the message flow to insert breakpoints. Click the desired flow.

The message flow will now be shown, just as in the flow development perspective. Breakpoints can be added by right-clicking connections and clicking **Add Breakpoint**. This can be done at any time with the view open, even during debugging.

The Flow Debug perspective is made up of four main views: Flow Debug, Flow Breakpoints, Flow Debug Message and Flow Development. The breakpoints view lists all the breakpoints on all flows; it shares its screen space with the message view which shows the message in its current state in a tree format.

Once a breakpoint has been added, any messages entering the flows containing breakpoints will be halted at the breakpoint. The toolkit is capable only of showing one instance of each flow at a time, so if two are running, the second will not be shown until the first is complete.

Figure 4-6 on page 92 shows the Flow Debug perspective. There are two breakpoints, each shown as a gray circle. Where the flow is currently halted, the gray circle is surrounded by an orange circle. The Flow Debug Message view (top right of the figure) shows the message in its current state. In this case, the message contains XML; this is also shown in the tree.

At the top of the Flow Debug view, the control buttons can be seen. These are how the debugging is controlled in conjunction with the flow development view. These will be familiar to anyone who has used a debugger before. The important buttons to note include Resume Flow Execution (third from the left), Run Flow To Completion (second from the right) and Step Into Source Code (far right), which opens the Debug perspective and allows debugging of the ESQL code (see Figure 4-7 on page 93).

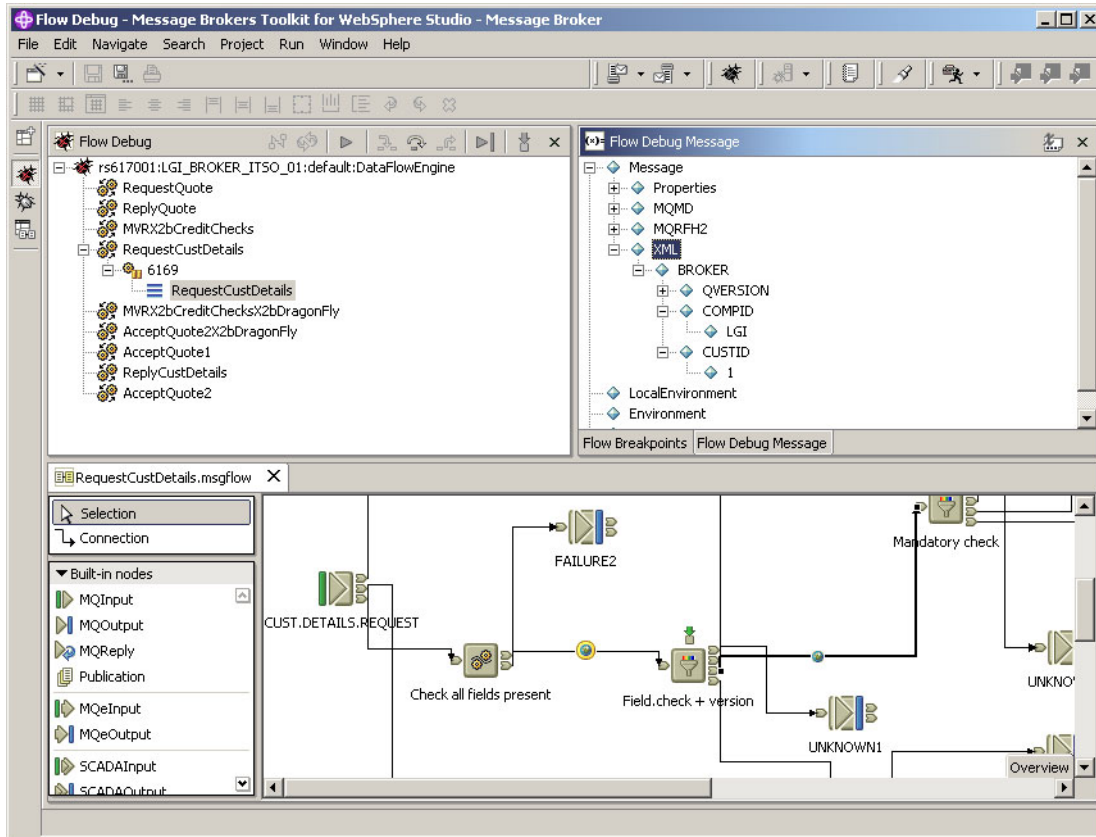


Figure 4-6 The Flow Debug perspective, during debugging of the RequestCustDetails flow

ESQL is debugged in much the same way as the message flow. Breakpoints are added by right-clicking the gray bar along the left side of the ESQL code and clicking **Add Breakpoint**. They can also be added, or removed, by double-clicking in the gray bar at the appropriate location. Breakpoints are now shown as blue/gray circles.

The top right panel of the Debug view contains four views. These are Variables, which shows all variables in the code along with the message in its current state, as visible to the ESQL code; Breakpoint, a list of all ESQL breakpoints; Expressions and Display, which are not relevant here and can be ignored.

Once breakpoints have been inserted into the ESQL, they are still active without the need to explicitly click the **Step Into Source Code** button. After execution of the ESQL is finished, the toolkit will automatically revert to the Flow Debug perspective, leaving the Debug perspective open in the background, and vice versa.

Similar to the Flow Debug view, the Debug view has buttons which control the debugging. These can be seen at the top of the Debug view as shown in Figure 4-7. Full descriptions of all these are given in the help section, but the main ones are Step Into (sixth from the left) which debugs the highlighted line of code and pauses at the next, Step Over (seventh from the left) which skips debugging of the highlighted line of code and pauses, and Run To Return (eighth from the left) which will execute the current method and then pause for further debugging. This final function will often return to the Flow Debug view, where a temporary breakpoint appears, to demonstrate where the code is currently paused, but the breakpoint will disappear once the flow is complete. Resume and Terminate are self-explanatory, with Resume continuing until either another breakpoint is encountered or the flow terminates.

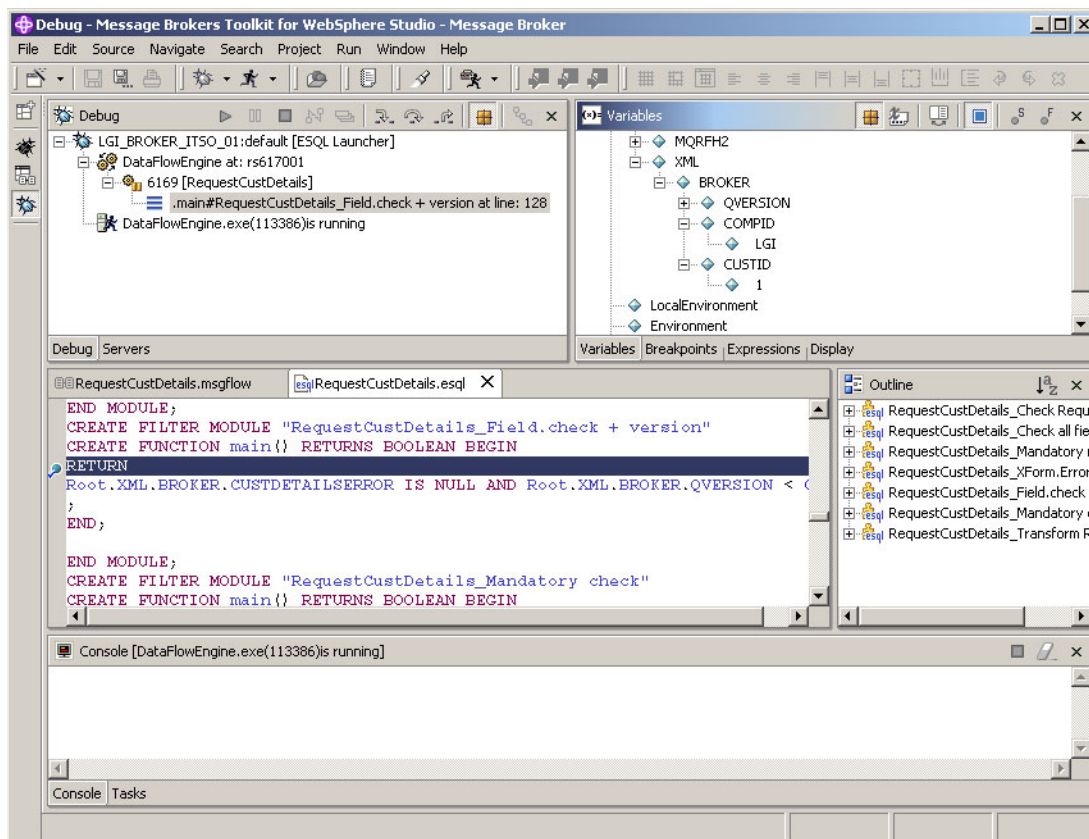


Figure 4-7 The Debug perspective showing ESQL being debugged

Note: Do not forget to detach the Debugger when you have finished debugging and before you do anything else.

4.6 IBM HTTP Server

Version: IBM HTTP Server V1.3.27

IBM HTTP Server can serve your Web content from WebSphere Application Server via the Web server plug-in. IBM HTTP Server has logs to use for problem determination on the Web server side.

access.log

The access.log, by default, records the accesses to the HTTP Server. You can see what is being requested and the response code.

error.log

The error.log contains information about the IBM HTTP Server status. It should tell you if the WebSphere plug-in was initialized.

Example 4-15 Initializing plug-in in the error.log

```
[Tue Nov 18 15:33:25 2003] [notice] Initializing the WebSphere Plugin  
[Tue Nov 18 15:33:27 2003] [notice] Instance name is Apache 2332
```

4.7 IBM DB2 UDB

Version: IBM DB2 Universal Database xV8.1

The key files required to assist with diagnosis of DB2 related problems are:

- ▶ db2diag.log file
- ▶ Trap files
- ▶ Dump files
- ▶ Messages files

These files are generated or updated when different events or problems occur.

db2diag.log file

The db2diag.log contains most of the key information used for DB2 problem diagnosis; this file is located in the DB2 diagnostic directory, defined by the DIAGPATH variable in the Database Manager Configuration.

By default, the directory is defined as follows:

- ▶ **UNIX:** \$HOME/sqllib/db2dump, where \$HOME is the DB2 instance owner's home directory
- ▶ **Windows or OS/2®:** INSTALL PATH\SQLLIB\<DB2INSTANCE>, where INSTALL PATH is the directory where DB2 is installed

The Database Manager Configuration also controls how much information is logged to the db2diag.log through the use of the diagnostic level, or DIAGLEVEL variable. The DIAGLEVEL can be set from 0 to 4, but this setting by default is 3, which is usually sufficient for most problems.

Trap files

Whenever a DB2 process receives a signal or exception (raised by the operating system as a result of a system event), a trap file is generated in the DB2 diagnostic directory.

The files are created using the following naming convention:

- ▶ **UNIX:** tpppppp.nnn, where pppppp is the process ID (PID) and nnn is the node where the trap occurred
- ▶ **Intel®:** DBppptt.TRP, where ppp is the process ID (PID) and ttt is the thread ID (TID)

Dump files

When DB2 determines that a serious problem, often related to the internal function of DB2, has been detected, a dump will often be taken and the file(s) will be located in the DIAGPATH directory. The filenames will either be pppppp.nnn or lppppppp.nnn for UNIX, or pppttt.nnn or lpppttt.nnn for Windows.

Messages files

Some DB2 utilities such as BIND, LOAD, EXPORT, and IMPORT provide an option to dump out a messages file to a user-defined location. These files contain useful information to report the progress, success or failure of the utility that was run and can assist with problem determination.

db2support

The db2support utility is designed to automatically collect all DB2 and system diagnostic information available (including information described in previous

pages). It has an optional interactive "Question and Answer" session available to help collect information for problems which you may want additional help investigating.

The db2support utility was made available in DB2 V7 (FP4) and is invoked as follows:

```
db2support <output path> -d <database name> -c
```

The output is collected and stored in db2support.zip.

4.7.1 Java Database Connector tracing

JDBC is the connector between the WebSphere for z/OS and the DB2 UDB for z/OS. This connector allows communication between the two products; requests to DB2 go through this connector, and this trace details what happens during this communication.

The JDBC trace is useful for diagnosing problems in the DB2 Structured Query Language for Java and Java Database Connector (SQLJ/JDBC). The output will go to an HFS file specified in the JDBC properties file.

JDBC trace information shows Java methods, database names, plan names, usernames, or connection pools.

4.7.2 Running a JDBC trace

The following steps describe the procedure to obtain the JDBC trace.

1. Set up the environmental variable parameter in the current.env file:

```
DB2SQLJPROPERTIES=/usr/lpp/DB2/DB2710/classes/wscdb_DB2sqljdbc.properties
```

2. In this properties file, called wscdb_DB2sqljdbc.properties, we set up the variable DB2SQLJ_TRACE_FILENAME to enable the SQLJ/JDBC trace and specify the name of the file to which the trace is written:

```
DB2SQLJ_TRACE_FILENAME=/tmp/IVP2_jdbctrace
```

Note: You can specify the path and file name that you want, but we use /tmp/IVP2_jdbctrace as an example, for clarity.

3. The JDBC trace produces two HFS files:

- /tmp/IVP2_jdbctrace

This file will be in binary format. Now you need to format it using the **DB2sqljtrace** command (as shown in the following step).

– /tmp/IVP2_jdbctrace.JTRACE

The second file contains readable text.

4. To format the binary trace data, we use the following **DB2sqljtrace** command in the USS environment (OMVS or telnet):

```
DB2sqljtrace fmt|flw TRACE_FILENAME > OUTPUT_FILENAME
```

Be sure, in the **PATH** and **LIBPATH** environmental variables, that there are JDBC path and libraries to run this command correctly.

You can change them with the following commands:

```
export PATH=$PATH:/usr/lpp/DB2/DB2710/bin
export LIBPATH=$LIBPATH:/usr/lpp/DB2/DB2710/lib
```

Note: The IBM default path is `/usr/lpp/DB2/DB2710/`, but you may have another path, depending on your installation.

You can verify that they are correct with the following commands:

```
echo $PATH
echo $LIBPATH
```

– **fmt**, **flw** commands

We can use the **fmt** subcommand or the **flw** subcommand, as follows:

- **fmt** - specifies that the output trace is to contain a record of each time a function is entered or exited before the failure occurs.
 - **flw** - specifies that the output trace is to contain the function flow before the failure occurs.
- **OUTPUT_FILENAME** is the name of the file where we want the new formatted trace.

4.8 IBM Tivoli Directory Server

Version: IBM Tivoli Directory Server V5.1

In the eMerge scenario, the directory server, Lightweight Directory Access Protocol (LDAP) server, is used as a user registry to perform user authentication.

WebSphere Application Server security provides and supports implementation of most major LDAP directory servers, which can act as the repository for user and group information. These LDAP servers are called by the product processes

(servers) for authenticating a user and other security-related tasks (for example, getting user or group information).

If the error logs do not provide enough information to resolve a problem, you can run the IBM Directory Server in a special debug mode that generates very detailed information. The server executable **ibmslapd** must be run from a command prompt to enable debug output. The syntax is as follows:

```
ldtrc on
ibmslapd -h bitmask
```

where the specified bitmask value determines which categories of debug output are generated.

Table 4-1 Debug categories, bitmask values

Hex	Decimal Value Description
0x0001 - 1	LDAP_DEBUG_TRACE Entry and exit from routines
0x0002 - 2	LDAP_DEBUG_PACKETS Packet activity
0x0004 - 4	LDAP_DEBUG_ARGS Data arguments from requests
0x0008 - 8	LDAP_DEBUG_CONNS Connection activity
0x0010 - 16	LDAP_DEBUG_BER Encoding and decoding of data
0x0020 - 32	LDAP_DEBUG_FILTER Search filters
0x0040 - 64	LDAP_DEBUG_MESSAGE Messaging subsystem activities and events
0x0080 - 128	LDAP_DEBUG_ACL Access Control List activities
0x0100 - 256	LDAP_DEBUG_STATS Operational statistics
0x0200 - 512	LDAP_DEBUG_THREAD Threading statistics
0x0400 - 1024	LDAP_DEBUG_REPL Replication statistics
0x0800 - 2048	LDAP_DEBUG_PARSE Parsing activities
0x1000 - 4096	LDAP_DEBUG_PERFORMANCE Relational backend performance statistics
0x1000 - 8192	LDAP_DEBUG_RDBM Relational backend activities (RDBM)
0x4000 - 16384	LDAP_DEBUG_REFERRAL Referral activities
0x8000 - 32768	LDAP_DEBUG_ERROR Error conditions
0xffff - 65535	ALL
0x7fffffff - 2147483647	LDAP_DEBUG_ANY All levels of debug



Problem determination tools: development environment

This chapter is an extension to Appendix 4, “Problem determination tools: runtime environment” on page 43. It focuses on problem determination tools in the runtime environment, primarily the WebSphere Studio Application Developer tools.

WebSphere Studio tools were looked at separately because they represent a large part of the general tools and because WebSphere Studio is the framework where tools will be converging in the future. WebSphere Studio tools also represent a different class, not only because they mainly come from the development environment, but also because they are “platform-independent” in the sense that the systems investigated by the tools can be on any platform. The tools can be used either during development and the application may be deployed on any platform, or they can assist in problem determination remotely on any platform and in any runtime.

5.1 WebSphere Studio

When you face problems with your application, you might have difficulty finding the errors during runtime just based on logs and trace. You may need to review application settings or code. You might also want to step through code execution or see statistics on your application. We will look at using WebSphere Studio Application Developer and WebSphere Application Studio Developer Integration Edition as tools for debugging application problems.

For more extensive help on developing and debugging with WebSphere Studio Application Developer and WebSphere Application Studio Developer Integration Edition, refer to the help documentation or the IBM Redbook, *WebSphere Studio Application Developer Version 5 Programming Guide*, SG24-6957-00.

If you have problems with WebSphere Studio itself, you can find some useful information in the *Finding and Interpreting WebSphere Studio Application Developer Log Files* article at:

http://www-106.ibm.com/developerworks/websphere/library/techarticles/0301_cartledge/cartledge.html

5.2 WebSphere Studio Application Developer Integration Edition V5.0.2

WebSphere Application Studio Developer Integration Edition was used to work with the eMerge scenario. WebSphere Application Studio Developer Integration Edition allows you to work with features available in WebSphere Application Server Enterprise such as Business Choreographer, extended messaging and many other features.

WebSphere Application Studio Developer Integration Edition has many features to help with problem determination for your application. Some of the same things available in WebSphere Application Server, like tracing and log viewing, are in WebSphere Application Studio Developer Integration Edition. There are more features available in WebSphere Application Studio Developer Integration Edition such as compiling, debugging, using the test environment, monitoring and profiling your application.

5.2.1 Application assembly

You can use WebSphere Studio Application Developer to assemble your EAR file. You can also review and change your application settings. This could include security settings, EJB settings, servlet and JSP mappings, JNDI names,

resource references and other settings. For instance, you may need to add user roles or change access intents on EJBs to conform to database specifications.

5.2.2 Compile

Your application should compile without errors in WebSphere Application Studio Developer Integration Edition. Before exporting your application or publishing to a test server, you should review the tasks view for problems related to your application. This will help prevent later problems with the application.

5.2.3 Component test perspective

The component test perspective lets you create different component testcases and run unit testing on different aspects of your application.

There are different types unit testcases that you can create and use:

1. Manual testcases

A manual testcase guides a user through a series of tasks. The user marks tasks as passed or failed.

2. HTTP testcases

With an HTTP testcase, you can run HTTP operations and queries against an HTTP Server.

3. Java testcases using JUnit

With JUnit, you create and run Java testcases.

5.2.4 IBM Agent Controller

The IBM Agent Controller allows WebSphere Studio Application Developer Integration Edition to talk to remote servers, log files and other resources. It is installed with WebSphere Studio Application Developer Intergration Edition. You can also install it as a stand-alone product on a remote machine. See your WebSphere Studio Application Developer Integration Edition installation guide for specific information.

5.2.5 Test environment

The test environment runs an internal version WebSphere Application Server so you can test your application while rapidly making changes and republishing to the test server. You can use the test environment while developing or if you need to review a finished application.

Application server

You can create test application servers in WebSphere Studio Application Developer Integration Edition to install and deploy your applications.

To create a test application server:

1. In the J2EE Hierarchy view, right-click the servers and select **New -> Server and Server Configuration**. Choose **Create a New Server and Server Configuration**.
2. Name the server and select a server type such as **EE Test Environment**.
3. After you create your server, expand Server Configurations and right-click the server you create. Select **Add -> <your_EAR_file>**.
4. Then open the configuration and configure your test server with resources and other settings that your application requires.

For more details on creating and working with test application servers, review the WebSphere Studio Application Developer Integration Edition help text.

Administrative Console

If you prefer to manage your test server with the Administrative Console installed with WebSphere Application Server, you can enable it on your test application server. Otherwise, there are configuration panels inside of WebSphere Studio Application Developer for creating resources, managing security, and other settings. For some features, you may have to use the Administrative Console since your settings may not be available in the WebSphere Studio Application Developer Integration Edition configuration panels.

To enable the Administrative Console:

1. Open your server configuration.
2. Switch to the Configuration tab.
3. Check **Enable Administrative Console**.
4. Save and restart your test server.
5. Access the Administrative Console using <http://hostname:port/admin>. The default port is 9090.

Universal Test Client

The Universal Test Client (UTC) is a Web interface test tool. It allows you to explore the JNDI namespace, interact with your EJBs, test methods and create instances. This is a good tool for unit testing your EJBs.

Follow these steps to enable and use UTC on your test server:

1. Go to the Server Configuration to open the server configuration.
2. Select the **Configuration** tab and check **Enable Universal Test Client**.
3. Save and restart your server.
4. In the Servers view, right-click the server and select **Run universal test client**. Or, to be taken directly to the correct area for an EJB:
 - a. In the Navigator view, right-click an enterprise bean resource.
 - b. Select **Run on Server**. The EJB page of the UTC opens and you can interact with the EJB.

With the UTC, you can verify that your EJBs are working correctly by directly accessing their methods and creating instances of your beans.

It is possible to point the JNDI Explorer of the Universal Test Client to an alternative server. This can be necessary when using a WebSphere Application Server cluster, for example. To do this, run the Universal Test Client, then click the **JNDI Properties** icon in the top right corner (second from right). This allows you to change the Provider URL and the Initial Factory. It also allows you to add custom name value pairs to the namespace.

It is also possible to install the Universal Test Client to a WebSphere Application Server. In C:\Program Files\IBM\WebSphere Studio\Application Developer\v5.1\wstools\eclipse\plugins\com.ibm.etools.utc_5.1.0, there is the IBMUTC.ear which can be deployed to WebSphere Application Server in the usual manner. It is then accessible at http://<server_name>/UTC. This can be useful in a production environment where the tooling is not installed.

Logs and trace

The test environment will produce all the normal logs of a WebSphere Application Server application server. You can view the JVM and service logs for information on problems with your application.

The JVM System.out log should appear by default in the Console window in the Server perspective.

You can also enable trace as you do with WebSphere Application Server. For specifics about using trace, see 7.1.6, “Tracing” on page 149.

To enable trace on your test application server:

1. Open your server configuration.
2. Select the **Trace** tab.
3. Check **Enable trace** and enter a trace string and a trace file.

4. Restart your server.

5.2.6 Debugger

The WebSphere Application Studio Developer Integration Edition debugger allows you to step through your application wherever you wish to watch values change and review the logic flow of your application. You can set up breakpoints to stop the application and walk through specific code. Your variables are all available for viewing while you debug. You can step over and into methods and isolate where your application problems are in the code. For instance, stepping through the code can help track down a mysterious null pointer exception by narrowing down exactly where the exception occurs.

Local

You can use the debugger on a local test server. You can set breakpoints when you want the application execution to stop. At the breakpoints, you can step through sections of code that are potentially causing problems.

To start the debugger on a test application server:

1. Open the Debug perspective.
2. Select **Run -> Debug**.
3. Select your test server and click **Debug**.
4. When the server has started, work with your application. At breakpoints, return to the debugger, step through code and examine variables.

For more details on working with the debugger, review the WebSphere Studio Application Developer Integration Edition help text.

Remote

Using the remote debugger is similar to using it locally, appearance-wise, because the debugger interface runs locally. The application server running the application and the debug engine run remotely. You need to have the IBM Agent Controller installed on the remote machine. You will need the code that you want to debug in your WebSphere Studio Application Developer Integration Edition repository and installed on a remote server.

The following are the basic steps needed to debug remotely. For more extensive information on using the debugger in WebSphere Studio, refer to the help documentation provided in WebSphere Studio.

1. Install IBM Agent Controller on a remote machine if not present.
2. Install the application EAR on the application server if not installed.

3. Import the EAR with the source code into WebSphere Studio Application Developer if not already present.
4. Set breakpoints in WebSphere Studio Application Developer where you want to stop in the code.
5. On your application server running remotely, set the configuration to use the debugging service. In the Administrative Console go to **Servers -> Application Servers** and select your server. Go to **Debugging Service** and check **Startup**. The default debugging port is 7777.
6. Next, under Additional Properties, select **Transaction Service**. Change the Total transaction lifetime timeout to 0 to disable the transaction timeouts. Also change the Client inactivity timeout to 0 and click **OK**. This prevents transactions from timing out as you stop to debug the code in WebSphere Studio Application Developer Integration Edition.
7. Restart the application server in debug mode.
8. In WebSphere Studio Application Developer Integration Edition, select **Debug** and choose a new WebSphere Application Server. The default port listed should be 7777. Change it if you changed it on your application server. Enter the hostname for the machine running the application server. The code you put breakpoints in should be listed as the source. Apply and select **Debug**.
9. Run your program and switch to WebSphere Studio Application Developer Integration Edition when you hit a breakpoint. You will be asked whether you want to step over or step into the code. Continue the debugging process.

5.2.7 Process debugger

The process debugger can be used to visually debug business processes. As with using the debugger, you can step through a business process node by node and step into or over the nodes. The process debugger also interacts with the regular debugger to review Java code. You can also change the input and output for the nodes in the process.

You can use the process debugger to find problems such as:

1. Incorrectly connected activities.
2. Incorrect conditional branches.
3. Infinite loops that should be finite.

The process debugger can be used locally or remotely. You need the IBM Agent Controller installed for both local and remote process debugging. You will also need the process engine running locally or remotely.

The following are general instructions to attach and use the debugger. You can use the process debugger with or without the Java debugger. For more details, refer to the WebSphere Application Studio Developer Integration Edition help files.

Using the process debugger

If you only want to debug a process and not the Java code, follow these directions.

1. Check that the IBM Agent Controller is running.
2. Check that the server on which you will debug is running.
3. Add breakpoints to your process by right-clicking the control links in your .process file.

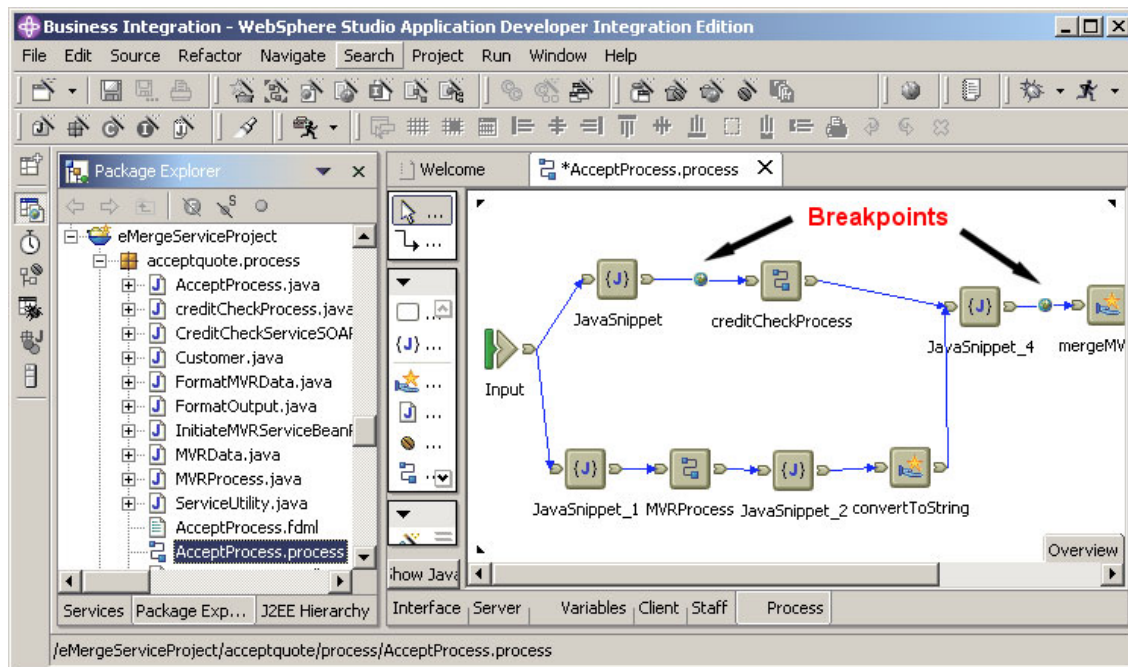


Figure 5-1 Adding breakpoints of the process

4. Open the Process Debug perspective.
5. Attach the process debugger by clicking the **Attach to Process Engine** icon.

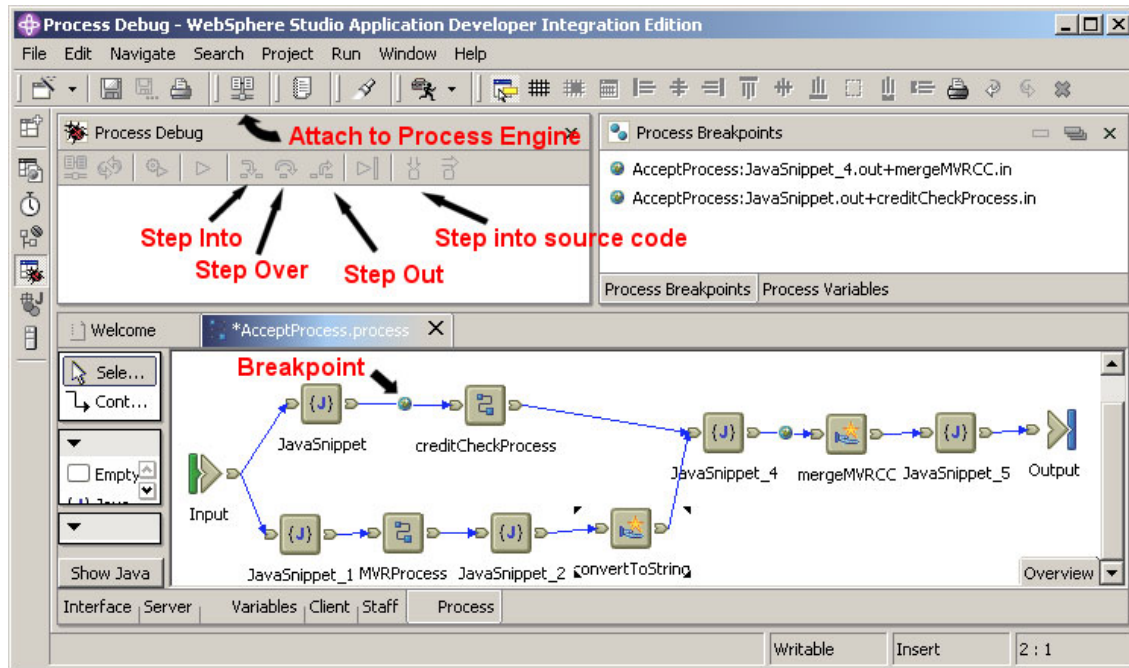


Figure 5-2 Process debug perspective

6. To attach a process, select an existing hostname or add a new one in the Attach to Process Engine window. Click **Next**.
7. Select the server process. Click **Finish**.
8. Wait for the Web client browser window to load.
9. Enter any input values and start the process.
10. Use the process debugger to step through the process by stepping over or into activities, blocks, loops, subprocess, and source code.
11. When you are finished, detach the process with the **Detach from Selected Process Engine** icon.

Using the process debugger with the Java debugger

If you want to use the process debugger and also step into Java code, follow these directions.

1. Add a breakpoint to the a Java snippet in your process.
2. Switch to the Server perspective with the servers view. Right-click your test server and select **Debug**.

3. When the server starts, switch back to the Process Debug perspective and click the **Attach to Process Engine** icon.
4. To attach a process, select an existing hostname or add a new one in the Attach to Process Engine window. Click **Next**.
5. Select the server process. Click **Finish**.
6. Wait for the Web client browser window to load.
7. Enter any input values and start the process.
8. Use the process debugger to step through the process by stepping over or into activities, blocks, loops, subprocesses, and source code. You will also be able to look at your Java code to step into or over and observe variables.
9. When you are finished, detach the process with the **Detach from Selected Process Engine** icon.

Using the process debugger remotely

Using the process debugger on a WebSphere Application Server Enterprise application server is similar to using the Java debugger remotely. You need to have the application installed on the remote server and in the workspace.

1. On your application server on WebSphere Application Server Enterprise with the application to debug installed, go to **Servers -> Application Servers** and select your server. Under Additional Properties, select **Debugging Service**. Check the **Startup** box and click **OK**.
2. Next, under Additional Properties, select **Transaction Service**. Change the Total transaction lifetime timeout to 0 to disable the transaction timeouts. Also change the clienter inactivity timeout to 0 and click **OK**. This prevents transactions from timing out as you stop to debug the code in WebSphere Studio Application Developer Integration Edition.
3. Restart your application server.
4. Open the Process Debug perspective and click the **Attach to Process Engine** icon. If the application server is running locally, select **localhost**, otherwise add the host for your application server.
5. Select the server process.
6. Use the process debugger as described in “Using the process debugger” on page 108.

5.2.8 Profiling perspective

With the profiling perspective, you can use viewing and monitoring tools to review your application and see how it is working.

Viewing and analyzing logs

With WebSphere Application Studio Developer Integration Edition, you can view the service log, usually named activity.log. Like Log Analyzer, which comes with WebSphere Application Server, you can view log messages and analyze them with the symptom database.

To import a symptom database to use with service logs in WebSphere Application Studio Developer Integration Edition, go to **File -> Import** and select **Symptom Database File**. Select either local or remote symptom databases to use with your log files.

To open the Log Analyzer with the Profiling perspective, open **Window -> Open Perspective -> Profiling**. Then import a log file to analyze. Go to **File -> Import** and select **WebSphere Application Server Log File**. Select either a local or remote machine where your log file is located and the name of the file to open. Then you can view and analyze messages in the log.

Monitoring

You can monitor an application while its running and view a variety of statistics about your application runtime. In the Profiling perspective, under the Profile menu, you can launch or attach a Java process, such as your application server, and monitor it. You can also attach to a remote process.

You can view statistics on various aspects of your application, including:

1. Package view
2. Class method view
3. Method view
4. Class instance view
5. Instance view

You can view graphical representations of different paths of your application, including:

1. Execution flow view
2. Object reference view
3. Method execution view
4. Method invocation view
5. Heap view

TCP/IP Monitor

If you want to view the TCP/IP traffic and monitor requests and responses between a Web browser and an application server involved in your application, you can enable TCP/IP monitoring. To enable it, go to **Preferences -> Servers -> TCP/IP Monitor** and select **Show TCP/IP Monitor view when there is activity**. You can only use this locally. You can open the TCP/IP Monitor view, which shows the request and responses.

Remote Agent Controller (RAC)

Remote Agent Controller (RAC) is a simple yet powerful daemon/service running on the Server where WebSphere is installed. It allows tools in WebSphere Studio and other applications to communicate with a remote WebSphere. It allows for instance remote profiling and a remote deploy for testing purposes.

IBM supports remote profiling on all platforms where RAC is supported, including remote deployment test on Windows and Linux (for the server) in WebSphere Studio 5.0.x and on Windows, AIX and Linux (for the server) in WebSphere Studio 5.1.x.

RAC can be found on the WebSphere Studio CD and has all RAC for all platforms. If a customer is using WebSphere Studio 5.0.x, we suggest you download RAC 5.0.2 from the following URL:

http://www3.software.ibm.com/ibmdl/pub/software/websphere/studiotools/html/501/wsad/install_rac.html

More information

You can find more information at the following URLs:

- ▶ Setting up a Remote WebSphere Application Server in WebSphere Studio V5:

http://www7b.boulder.ibm.com/wsdd/techjournal/0303_yuen/yuen.html

- ▶ Setting up a Remote WebSphere Test Server for Multiple Developers:

http://www7b.boulder.ibm.com/wsdd/library/techarticles/0303_karasiuk/karasiuk.html

Data to collect for support

To improve the response time, make sure you collect all the information listed below. IBM support will ask for these items.

- ▶ What is the OS system on the remote machine?
- ▶ If RAC is installed, attach the servicelog.log and serviceconfig.xml from the remote machine.

- ▶ If you have a problem connecting or executing, attach loggingUtil999.log and .log from the <Workspace_location>/metadata.
- ▶ What is the exact error message?
- ▶ Attach a screen capture of the message and the steps you took.
- ▶ Attach a screen capture of the steps you are taking to connect.
- ▶ Attach a screen capture of the remote server configuration in the tool; select the **Server** tab.
- ▶ Attach a screen capture of the remote file transfer artifact that should appear under the server folder in the tool.
- ▶ Check to see if there is a firewall or any software that would prevent the RAC to connect back to the WebSphere Studio Application Developer machine.

5.3 WebSphere Studio Application Developer V5.1

WebSphere Studio Application Developer 5.1 includes all the non-WebSphere Application Server Enterprise related problem determination tools in WebSphere Application Studio Developer Integration Edition, with some additions.

WebSphere Studio Application Developer 5.1 is based on Eclipse 2.1.1 which provides better tool performance and usability. There is support for a Hot Method replace that allows you to update code while debugging. It also includes Active Script debugging and SQLJ debugging.

5.3.1 Log and Trace Analyzer for Autonomic Computing

The Log and Trace Analyzer for Autonomic Computing allows developers to combine log and trace files for several products into a single Eclipse-based interface, allowing the developer to combine pieces of information from several sources and correlate them by timestamp to determine the source of the error.

The need to combine the information from several sources is greatly increased as the number of components that make up the topology of an application increases. Having each product produce its own trace format requires developers to open and compare the debug information from several sources independently.

You can download this tool from alphaWorks. With the Log and Trace tool, a developer is allowed to import log and trace files, which are then passed through filters and converted to a common format for ease of viewing and interpreting.

Some of the capabilities of the tool are as follows:

1. Combine the WebSphere Log Analyzer and WebSphere Studio Profiling tool into a single interface.
2. Import different log files from their original format and map it to a common base event format.
3. Plug in and select a log file parser to import log files in a single view.
4. Display the set of imported log files, showing the relationships among events in the logs.
5. Correlate events in multiple log files into a correlation view.
6. Plug in and select correlation methods that determine those relationships.
7. Create, import, and export a symptom database for problem determination.
8. Analyze multiple log files based on multiple symptom databases.
9. Plug in a different analysis engine to perform problem determination on the log files entries.
10. Attach to a running JVM to collect log information using JSR47 or Apache Commons Logging APIs.
11. Profile a JAVA application and display the results in multiple views.
12. Attach to a running JVM to collect profiling and tracing information.

How to obtain and install the tool

You can download this tool from:

<http://www.alphaworks.ibm.com/tech/logandtrace>

Upon download, extract the corresponding file to a local directory. To run the application, execute the following command:

```
$TOOL_HOME/eclipse/ac.bat
```

The current image of the tool comes with Eclipse V2.1.1 so you will not need to download the Eclipse workbench.

Usage Scenarios

There are many situations in which a user could make use of the Log and Trace Analyzer for Autonomic Computing tool.

Among some of the scenarios are:

1. When running the Application Server, an administrator will need to look up an explanation of an error code received. To do this, the error log is analyzed

against a symptoms database that will provide the user with problem analysis.

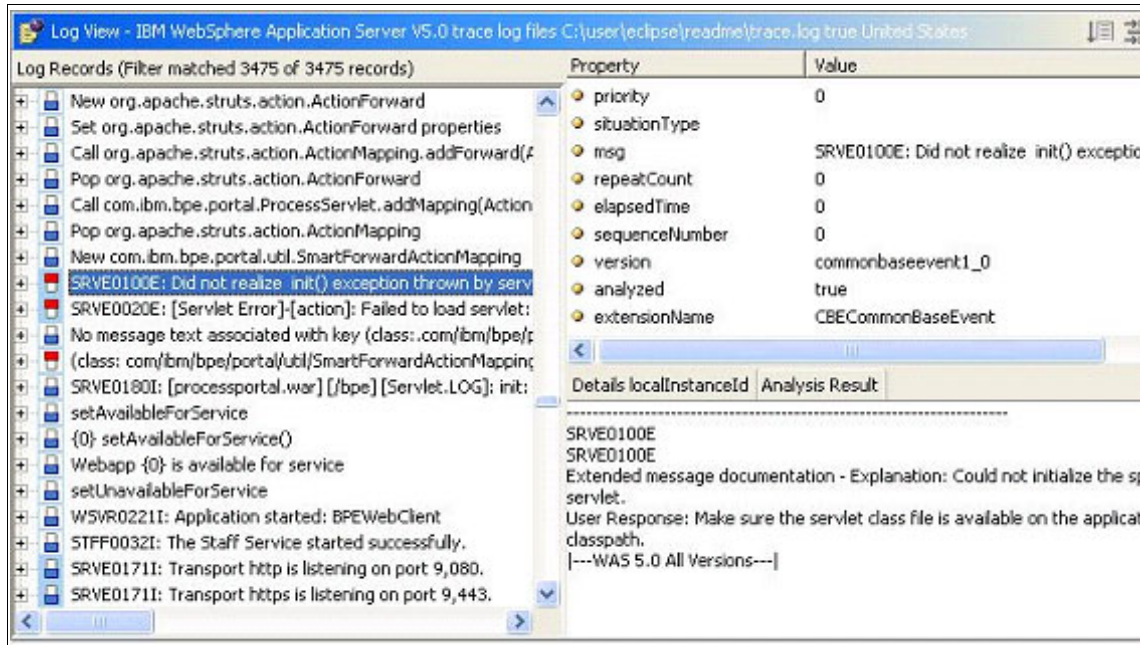


Figure 5-3 Analyzing the logs

2. If a CMP EJB is returning an error, a developer might need to verify whether the error is being reported on the server side or received from the database. Using the tool, the developer will import the server and database trace or log files into the trace project. Upon doing so, the developer will be able to correlate the two files and follow the execution of the application by timestamp.
3. A developer can also develop a parser for application log files following the Common Base Event model and correlate a user-specific log or trace file side by side with a WebSphere Application Server trace file. For example, a Business Partner has an application that connects a ERP program to an Application Server. If a parser is written for the log or trace files generated by the ERP program, these files can be compared side by side with those of the application server, thus facilitating the problem determination of the application across multiple products.
4. You are receiving errors from the HTTP Server and need to examine the access and error logs to determine the source of the problem. As shown in the following figure, with the Log and Trace tool, it becomes easy to correlate multiple log files into a single interface.

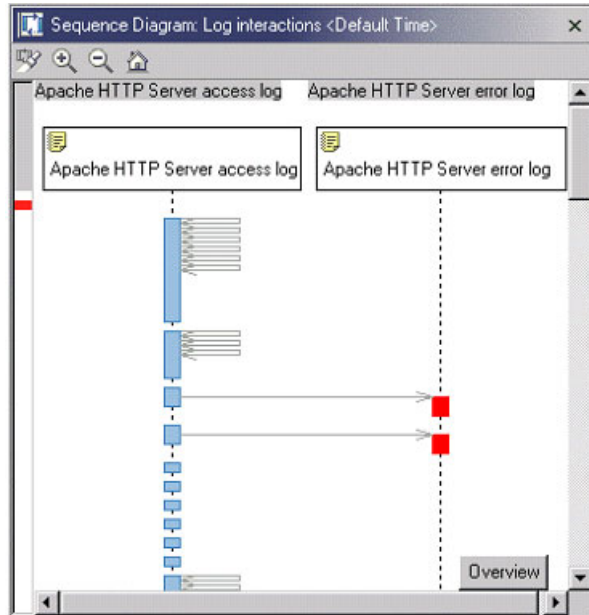


Figure 5-4 Sequence diagram

Usage tips

Maintain the trace files as small as possible to improve parsing performance. If possible, generate a new trace file every time a new problem is noted, backing up previous trace files.

Keep symptoms database updated. You can achieve this by downloading a new copy of the database before performing the analysis. Also, generate a local version which includes customized error problem descriptions.

5.3.2 J9 JVM: Hot Method replace

WebSphere Studio Application Developer 5.1 comes bundled with a J9 JVM in addition to the Sovereign JVM, which is the default for non-debug operations. The J9 JVM provides improved performance over the Sovereign JVM, thus making it a good choice when debugging applications such as JSPs. The main benefits of the J9 JVM are the added support for Hot code replace, which gives you the ability to make changes to running objects in a JVM, and Full Speed Debug, which allows developers to debug applications with JIT enabled.

In order to enable J9 support, the **Enable hot method replace in debug mode** from the server configuration panel must be selected.

The J9 JVM is recommended for debugging applications, but its use is not recommended in a production environment.

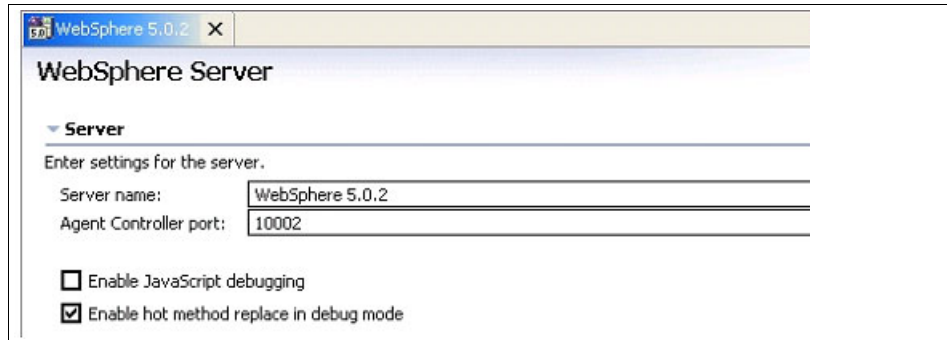


Figure 5-5 Enable hot method replace

5.3.3 Active Script debugging

With Active Script debugging, you can perform client side debugging of the following file types:

- ▶ Script:
 - JavaScript, or files with extension .js.
 - VisualBasic script, or files with extension .vbs.
- ▶ HTML files (.htm or .html) with the following types of embedded script:
 - JavaScript
 - VisualBasic script
- ▶ JSP files (.jsp) with the following types of embedded script:
 - JavaScript
 - VisualBasic script

The debugger attaches to the Web browser to step through scripts execution and breakpoints. In addition, a developer can monitor variables of the debugged application.

5.3.4 SQLJ debugging

The SQLJ debug adapter allows you to begin a debug session for SQLJ (or Java embedded with SQL). The debugger displays SQLJ source instead of a generated Java source.

Debug the configuration used to launch the SQLJ application. The SQLJ debug adapter offers the following accessibility features:

1. Visual focus indicators via cursors in editable objects and highlighted buttons, menu items and other selections.
2. Tooltip help for buttons and other selections.
3. Complete assistive technology enablement in wizards and dialogs.
4. Messages, dialogs, and wizards that persist until you close them.
5. Documentation that includes hover-over image descriptions and marked table headers.

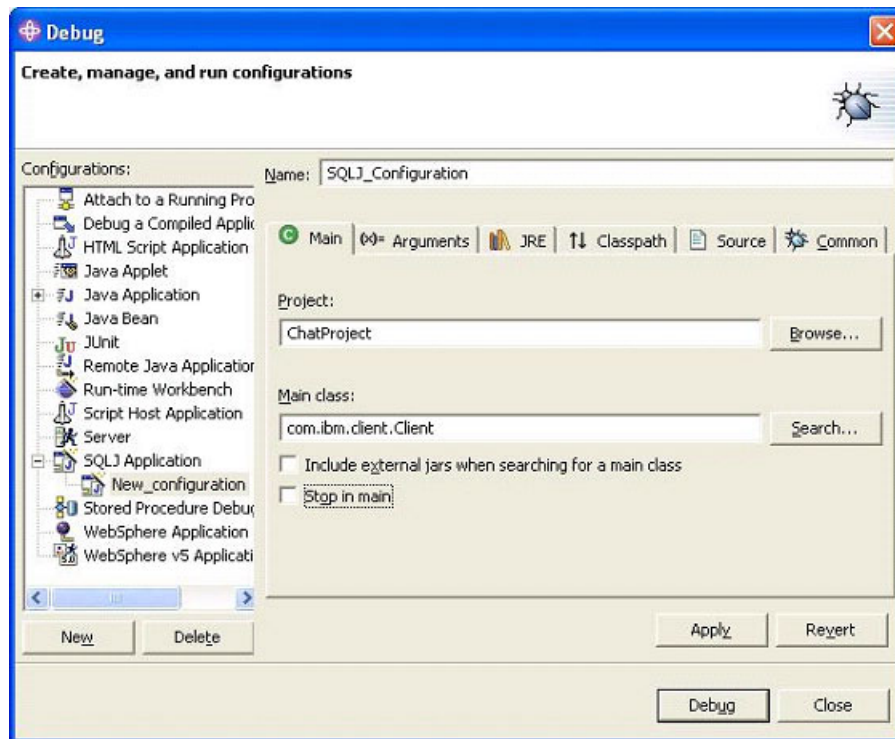


Figure 5-6 SQLJ debugging

5.3.5 J2EE Code Validation technology preview tool

The J2EE Code Validation tool is a technical preview that performs static analysis of your code to find problems. You can use this tool during development or on misbehaving Web modules to locate code problems.

Visit

http://www-106.ibm.com/developerworks/websphere/downloads/j2ee_code_validation.html to download the J2EE Code Validation tech preview. After you install the tool, follow these steps to perform validation.

1. Import your application into the WebSphere Studio Application Developer workspace.
2. Go to **Windows -> Preferences** and select **Validation**. Check **J2EE Code Validator** on the right. Click **OK**.
3. On the project that you want to validate, right-click and select **Properties**. Select **Validation** and select only **J2EE Code Validator**. Click **OK**.
4. Right-click again on the project that you want to validate and select **Run Validation**.
5. Review the Status window for analysis steps. Then review the Tasks window for task items listed. You can right-click a task item and select **Explain** for more details. You can double-click to open the source code.

5.4 RFHUTIL

RFHUTIL is a utility that you will find extremely helpful when diagnosing MQ transmission and data problems. This utility lets you read from a file, write (PUT) to queue, write to a file, read (GET) from a queue and perform BROWSE functions. This utility lets you interrogate the data and the MQ headers of the message, for example MQMD and RFH.

RFHUTIL allows test messages to be captured and stored in files, and then used to drive WebSphere MQ Integrator V2 applications. Output messages can also be read and displayed in a variety of formats. The formats include two types of XML as well as matched against a COBOL copybook. The data can be in EBCDIC or ASCII. An RFH2 header can be added to the message before the message is sent.

The RFHUTIL utility is not part of the WBI product; SupportPac™ IH03 is downloadable from the WebSphere MQ Family SupportPac site as follows:

<http://www-3.ibm.com/software/integration/support/supportpacs/individual/ih03.html>

The primary window for RFHUTIL is as follows:

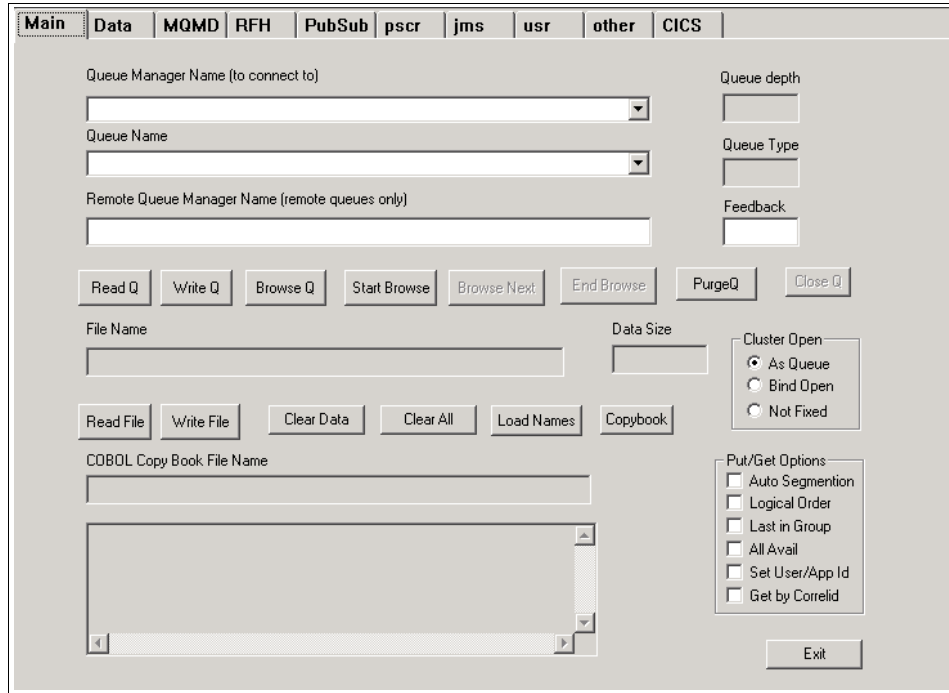


Figure 5-7 RFHUTIL utility application

An example of reviewing the data that has been read from the file is as follows. You can choose the data format for the display.

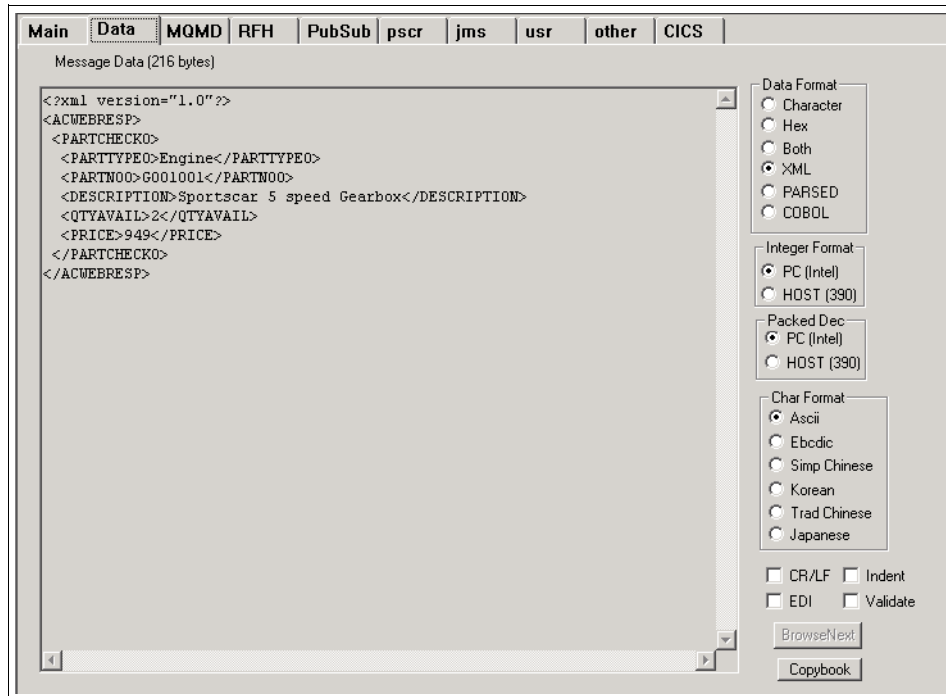


Figure 5-8 RFHUTIL



WebSphere Application Server : plug-in

In this chapter, we will focus on the determination of plug-in related problems. However, in the first steps of problem determination, it is usually not obvious whether the problem appears in the Web Server, the plug-in or after the request is successfully passed to the WebSphere Application Server. To assign the problem to one of those components, we propose a simple test in “Things to check” on page 126, which should help you determine which component is failing. In some cases, it is difficult to distinguish between problems that originate in the plug-in and problems caused by Web server malfunction. Therefore, some Web server related problems are also covered in this chapter.

6.1 Web server plug-in

In our scenario, each HTTP request that goes from the customer to the HTTP Server is processed by the plug-in, which is installed on the HTTP Server's machine.

The plug-in is based on the HTTP request, the workload and the status of WebSphere Application Server(s) and its configuration selects the Web container running on one of the WebSphere Application Server(s) to serve the request or route it back to the HTTP Server.

Request-processing algorithm

To determine the problem source, it is important to understand how the request is managed by a particular component. Therefore, a short description of plug-in request handling using a flow chart is called for.

The Web server plug-in runs inside the Web server process and acts as an intermediary between the HTTP Server and WebSphere. The plug-in decides whether the request is going to be served by WebSphere based on the requested URL and the port number on which the request was sent. The decision is made based on rules written in the plug-in XML configuration file, `plugin-cfg.xml`. If the request will be served by WebSphere, the plug-in will route it to the appropriate Web container.

We will use some of the plug-in's variables to describe how the plug-in works. A complete description of plug-in variables can be found in the WebSphere Application Server Info Center.

If the plug-in is installed and configured properly, it should work as shown in the flow chart shown in Figure 6-1 on page 125.

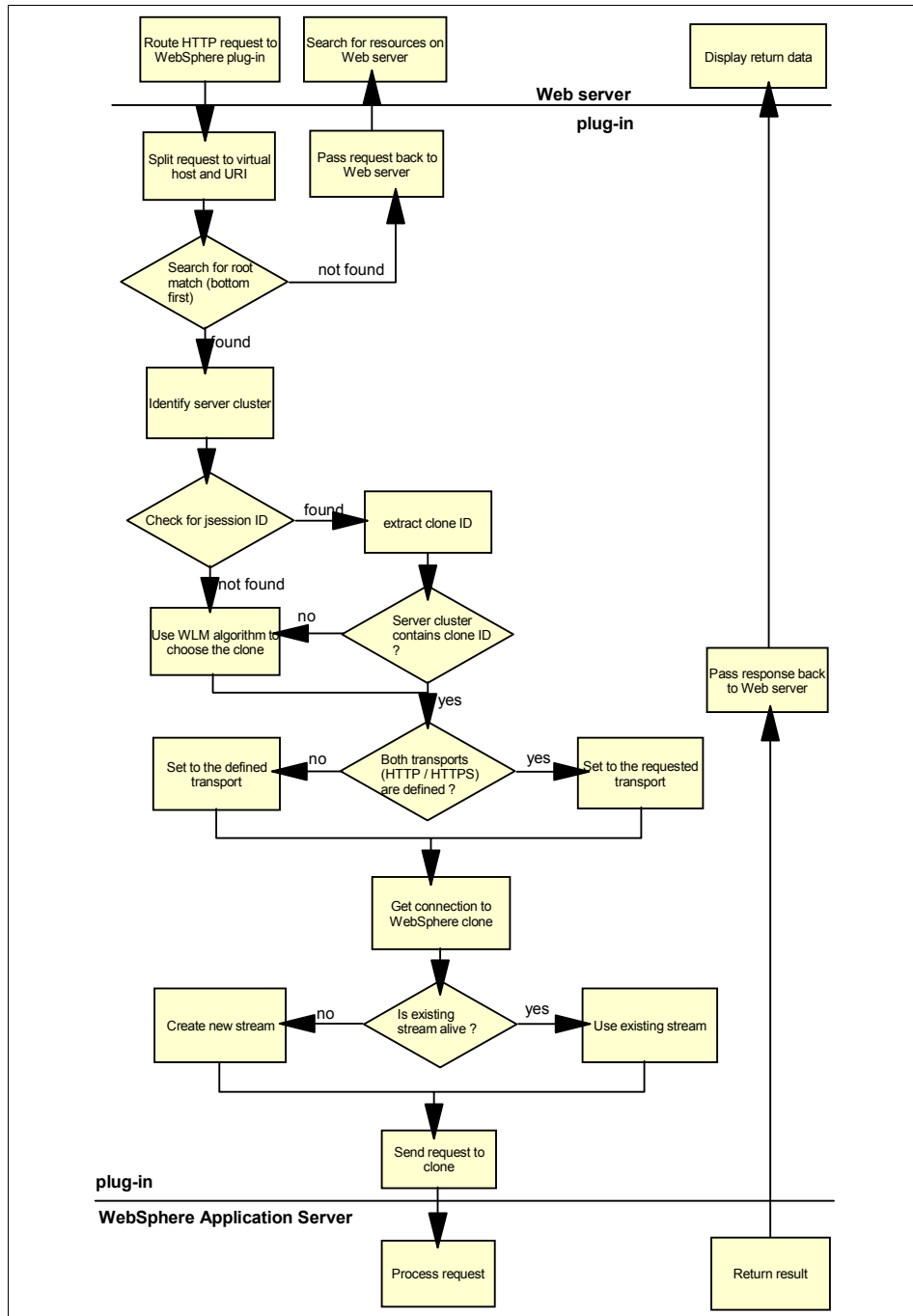


Figure 6-1 Plug-in HTTP request processing

1. The plug-in receives a request from the Web server and splits it between the VirtualHost and the URI.
2. The plug-in looks at each <Route> defined in the plugin-cfg.xml from the bottom up. It tries to match the request with the <VirtualHostGroup> and <UriGroup> defined for that particular <Route>. If the <Route> does not have definitions for <VirtualHostGroup>, the <Route> will match any hostname and port. The same applies to the <UriGroup>; if the <Route> does not have definitions for <UriGroup>, any URI will match.
3. If the plug-in was not able to find a match between the HTTP request and the <Route> nodes, the request is passed back to the HTTP Server which will handle the request by trying to find resources on the Web server.
4. As soon as the first <Route>, from the bottom up in the plugin-cfg.xml, matches the HTTP request, the plug-in marks the <ServerCluster> that is going to be used to process this request.
5. The plug-in checks the HTTP request URL or cookie for the JSESSIONID parameter. If it is found, the plug-in parses it and extracts the cloneID.
6. The plug-in checks to see if the <ServerCluster> contains the <Server> identified by the extracted cloneID.
7. If the server's cloneID matches the requested cloneID, this particular clone will process the request. Otherwise, the plug-in will choose the clone based on the <LoadBalance> and <LoadBalanceWeight> parameters.
8. The plug-in decides whether HTTP or HTTPS is going to be used to pass the request to the Web container based on the <Transport> defined in the plugin-cfg.xml and the requested transport. If both transports are defined in the <Transport>, the plug-in will use the original transport; otherwise, the defined transport will be used.
9. At this point, the plug-in sends a request to the Web container. If there is no response then the plug-in marks the <server> as down and the request is sent to another Web container based on the selection policy. The plug-in tests all <Server> nodes marked as down after the <RetryInterval>. By default, this is set to 60 seconds.
10. At this point, the plug-in waits for the response from the Web container. When the Web container replies, the plug-in sends that response to the Web server.

Things to check

Because the plug-ins and the Web server's logs are going to provide you with useful information in the problem determination process, you should locate the logs' names and locations in your environment.

In the configuration file for your Web server, check for the names and locations of the Web server's logs and the location of the plug-in configuration file

(plugin-cfg.xml), which configures the plug-in for the Web server. For details on your particular Web server's logging and plug-in definitions, consult your Web server's documentation.

In plugin-cfg.xml, check for the exact name and location of the plug-in log file. The plug-in log name and location is specified under the tag <Log> in the variable Name. As shown in Example 6-1, by default the plug-in writes to the http_plugin.log file in the <WebSphere_root>/logs/ directory.

Example 6-1 Plug-in logging location and file name determination

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Config ASDisableNagle="false" IISDisableNagle="false"
IgnoreDNSFailures="false" RefreshInterval="60" ResponseChunkSize="64">
  <Log LogLevel="Error"
Name="/usr/WebSphere/AppServer/logs/http_plugin.log"/>
  .....
</Config>
```

In addition to the name of the plug-in log file, the <Log> tag also defines the amount of information that will be logged. The amount of log data is specified by the variable LogLevel. Error, Warn, and Trace are the acceptable LogLevel parameters. Error is the default value, and logs the least data. Because of the performance impact of Trace level logging, it should only be used when trying to diagnose a problem.

Note: When you regenerate the plugin-cfg.xml file, the LogLevel is automatically reset to Error.

If you are not sure whether the plug-in was initialized correctly, you should check http_plugin.log. The following example shows records in http_plugin.log which indicate that the plug-in was successfully initialized.

Example 6-2 Successful plug-in initialization

```
[Mon Dec 1 13:55:15 2003] 000041ca 00000001 - PLUGIN: Plugins loaded.
[Mon Dec 1 13:55:15 2003] 000041ca 00000001 - PLUGIN:
-----System Information-----
[Mon Dec 1 13:55:15 2003] 000041ca 00000001 - PLUGIN: Bld version: 5.0.0
[Mon Dec 1 13:55:15 2003] 000041ca 00000001 - PLUGIN: Bld date: Jun 23 2003,
00:56:03
[Mon Dec 1 13:55:15 2003] 000041ca 00000001 - PLUGIN: Webserver:
IBM_HTTP_SERVER/1.3.26.2 Apache/1.3.26 (Unix)
[Mon Dec 1 13:55:15 2003] 000041ca 00000001 - PLUGIN: Hostname = m10d9ffd
```

```
[Mon Dec 1 13:55:15 2003] 000041ca 00000001 - PLUGIN: NOFILES = hard:
INFINITE, soft: 2000
[Mon Dec 1 13:55:15 2003] 000041ca 00000001 - PLUGIN: MAX COREFILE SZ = hard:
INFINITE, soft: 1073741312
[Mon Dec 1 13:55:15 2003] 000041ca 00000001 - PLUGIN: DATA = hard: INFINITE,
soft: 134217728
[Mon Dec 1 13:55:15 2003] 000041ca 00000001 - PLUGIN:
```

Connecting to an embedded HTTP Server

In the case where requests get to the Web server, but are not passed to the WebSphere Application Server, the first thing to check is whether you are able to connect to the Web application directly through the embedded HTTP Server. To do this, connect to WebSphere with your browser, using the port specified in plugin-cfg.xml under the server tag <Transport>. Shown in Example 6-3 is a segment of plugin-cfg.xml; by default, the Web container listens on port 9080.

Example 6-3 Listening port of Web container

```
<Server Name="server1">
  <Transport Hostname="websphere.server.domain" Port="9080" Protocol="http"
/>
  ...
</Server>
```

In this case, the direct URL for connecting to the embedded HTTP Server is:

<http://websphere.server.domain:9080/<URI>>

If you are not able to access the application using the Web container's listening port, you should check to see if you have a firewall between the browser and the HTTP Server and whether the Web container's listening port is blocked. In this case, you can either use the browser which is behind the firewall or temporarily open the port. The problem is related to the WebSphere application; refer to Chapter 7, "WebSphere Application Server: base" on page 135 for details.

If you can access the application only through the Web container's listening port, the problem relates either to the plug-in or the HTTP Server. To determine if the plug-in is failing, you should check the plug-in log file http_plugin.log; look for any error or warning messages.

For Web server related problems, review your Web server's error and access logs. The logs are stored in different files for each Web server:

- ▶ IBM HTTP Server and Apache: access.log and error.log
- ▶ Domino® Web server: httpd-log and httpd-error

- ▶ iPlanet: access and error
- ▶ Internet Information Services: timedatestamp.log

If this does not help you determine the problem, you should set the log level of the plug-in to Trace:

1. Edit plugin-cfg.xml and set the property LogLevel to Trace.
2. Reproduce the problem and check http_plugin.log.

The flow chart in Figure 6-1 on page 125 should help you to localize the problem source.

In the following sections, you will find problem determination details related to typical symptoms caused by a plug-in malfunction.

Symptom: The HTTP Server does not start

If the HTTP Server does not start, you should check the following:

1. Whether the plug-in is installed correctly.
Check the WebSphere installation log file <WebSphere_root>/log.txt for plug-in installation errors.
2. Whether the HTTP Server is installed correctly. Check the documentation for your HTTP Server.
3. Whether there are any syntax errors in the plugin-cfg.xml file.
Syntax errors in the plug-in configurational file prevent the HTTP Server from loading the plug-in properly. Because the error appears when the HTTP Server is loading, the plug-in indication for a syntax error in the plugin-cfg.xml file can be found in the HTTP Server's error log:

```
ws_common: websphereUpdateConfig: Failed parsing the plugin config file
```
4. Whether you have copied the plug-in configuration file (plugin-cfg.xml) from another location; change directories to match your local structure.
5. Whether the user name that the HTTP Server runs under has read access to the plug-in configuration and plug-in executable files.
The same Failed parsing the plugin config file message will be registered in the HTTP Server's error log.

Symptom: Internal Server Error (HTTP Error 500)

The most common causes for Internal Server Errors are the following:

1. WebSphere Application Server is not running.
2. The plug-in is not communicating with the WebSphere Application Server.

- a. Verify that the path you are requesting is defined as a URI in plugin-cfg.xml.
- b. If you are using a firewall between the HTTP Server and WebSphere Application Server, make sure that the ports specified in plugin-cfg.xml are opened in the firewall.
- c. Try to connect to the Web application directly through the embedded HTTP Server. You can do this by sending the HTTP request to the port on which the Web container is listening (by default, this is 9080). If you are not able to access the Web application using the Web container's listening ports, this indicates a problem involving your application server.
- d. Verify that the hostname in plugin-cfg.xml resolves properly from the HTTP Server machine. Use the command `nslookup <hostname>`. If the hostname does not resolve, replace it with the IP address.

Symptom: The plug-in behavior does not reflect changes in the plug-in configuration

Usually, changes in the plug-in configuration in the Administrative Console do not entail changes in the plug-in behavior because the configuration has to reload after the changes are made.

It is possible that the configuration was not reloaded because of syntax errors in the changed configuration. Check http_plugin.log for errors. For a detailed description of plugin-cfg.xml, refer to the WebSphere InfoCenter.

If the HTTP Server is running on a separate machine and the plug-in was regenerated from the Administrative Console, the plugin-cfg.xml must be copied over to the Web Server.

Symptom: Session affinity problems

The plug-in can manage session affinity for the servers in the same <ServerCluster>. If servers are not in the same <ServerCluster>, then session affinity could be implemented using clustering tool such as Load Balancer.

The Session Manager may use cookies, URL rewriting or SSL as session tracking mechanism. Based on the mechanism that is used to implement HTTP session affinity, you should check the following:

- ▶ If you are using cookies for HTTP session affinity, make sure that cookies are flowing between WebSphere Application Server and the browser:
 - Check that cookies are enabled on the browser.
 - Make sure that an appropriate session tracking mechanism is enabled in WebSphere Application server.

- Make sure that general parameters for HTTP session affinity are set correctly to accept the cookies.
- ▶ If you are using URL rewriting instead of cookies:
 - Ensure that there are no static HTML pages on your application's navigation path.
 - Ensure that your servlets and JSP files are implementing URL rewriting correctly.
- ▶ If you are using SSL as a session tracking mechanism:
 - Ensure that you have enabled SSL on the Web server. If you have a clustered environment, ensure that session persistence is enabled.
 - Verify that each <Server> specified in plugin-cfg.xml has its own unique CloneID. If a CloneID is not specified in the <Server> then session affinity will not be enabled for this particular server.

Symptom: Workload management problems

The plug-in can be used for HTTP workload management in order to serve requests from the specific HTTP Server and distribute them to a clustered application server environment.

If HTTP requests are not distributed to all servers, you should check the following:

- ▶ Whether you are able to connect from the HTTP Server to each WebSphere machine. If you are using a firewall between the HTTP Server and WebSphere Application Server, you might have to configure the firewall to enable HTTP requests and responses to flow between them.
- ▶ Whether you have all the servers that are used in the workload management scenario listed as <PrimaryServers> in the plug-in configurational file plugin-cfg.xml.

If you do not have a <PrimaryServers> list defined, the plug-in will perform workload management across all servers defined in the <ServerCluster>, if affinity has not been established.

The plug-in will perform workload management across all servers that are defined in the <PrimaryServers> list if affinity has not been established. If affinity has been established, the plug-in will not perform any workload management and will send the request directly to that server.

- ▶ Whether you receive a response from the server when you bypass the plug-in by specifying the Web container's listening port in the request.
 - Check plugin-cfg.xml if the server is listed in the <PrimaryServers> list.

- Check the load balancing type (<LoadBalance>) that is used. If the Random type is used, you cannot predict when a particular server will receive a request. By default, Round Robin is used.
- Check the plugin-cfg.xml parameter <RetryInterval>. If the server was recently restarted and the <RetryInterval> has not yet been reached, the plug-in can still mark the server as down.
- Check the plug-in parameter <LoadBalanceWeight> which is used for weighing servers for load balancing. The algorithm for this attribute decrements all weights within the server cluster until all weights reach zero. Once a particular server's weight reaches zero, no more requests are routed to that server until all servers in the cluster have a weight of zero. After all servers reach zero, the weights for all servers in the cluster are reset and the algorithm starts over. If the ratio between LoadBalanceWeight parameters is high, it could happen that one server already has its weight variable set to zero, because it has already served all requests, while others still have to serve a predefined number of HTTP requests. In such a case, the server with weight variable 0 will not receive any requests until all the others servers have served a predefined number of requests.
- ▶ Whether you receive a response from the server if you bypass the plug-in by specifying the Web container's listening port in the request; if you do not, the most common cause is that the server is not running.

Symptom: Error 404 Page Not Found

Usually, this error is caused by a misconfiguration in the plugin-cfg.xml

1. If you are trying to access a newly deployed application and receive Error 404, you should regenerate the plug-in and restart the HTTP Server.
2. Verify that the requested URI is defined in plugin-cfg.xml and that it is a member of the <UriGroup> defined under the <Route> item.

If SSL is used, you can get Error 404 if the plug-in cannot find the Global Security libraries.

Symptom: Unable to serve WebSphere resources over SSL

If you cannot reach SSL resources but unsecured resources are accessible, you will typically find one of the following error messages in http_plugin.log:

- ▶ lib_security: loadSecurityLibrary: Failed to load gsk library

The GSK was not installed or the installation is corrupt. You can determine if the GSK was not installed by searching for the file gsk5ssl.dll on all drives in Win32 or see if there are any libgsk5*.so files in /usr/lib on Unix. Try

reinstalling the plug-in to see if you can get the GSK to install in order to fix this.

- ▶ `ws_transport: transportInitializeSecurity: Keyring wasn't set`
The HTTPS transport defined in the configuration file was prematurely terminated and did not contain the Property definitions for the keyring and stashfile. Check your XML syntax for the line number given in the error messages that follow this one to make sure the Transport element contains definitions for the keyring and stashfile before it is terminated.

Symptom: Unable to serve static content

If your HTTP Server appears to be functioning correctly and the application server also works on its own, but browser requests for pages sent to the HTTP Server are not being served, this indicates that a problem is likely to appear in the WebSphere Application Server plug-in.

To check that the problem does not appear after a request is successfully passed to WebSphere Application Server, send a request to the embedded HTTP Server as described in “Connecting to an embedded HTTP Server” on page 128.

If the direct request to the embedded HTTP Server does not return the requested page, this indicates that the problem occurs after WebSphere receives the request.

If the page is served only when the request is sent directly to the embedded HTTP Server, continue with the following steps:

1. Determine whether the HTTP Server is attempting to serve the requested resource itself, rather than forwarding it to the WebSphere Application Server.
Check the HTTP Server access log. It may indicate that it could not find the file in its own document root directory.
2. Check the `plugin-cfg.xml`. It determines which requests sent to the HTTP Server are forwarded to the WebSphere Application Server, and to which application server.
 - a. In the WebSphere Application Server Administrative Console, expand the Environment tree control.
 - b. Select **Update WebSphere Plugin**.
 - c. Stop and restart the HTTP Server, then retry the Web request.
3. Check the `http_plugin.log` for clues to the problem.
4. Turn on plug-in tracing by setting the `LogLevel` attribute in the `plugin-cfg.xml` file to `Trace` and reloading the request, then check the `http_plugin.log` file. You should be able to see the plug-in attempting to match the request URI with the various URI definitions for the routes in the `plugin-cfg.xml`. You should also be

able to see what rules the plug-in is not matching against and decide whether you need to add new ones. If you just recently installed the application, you may need to manually regenerate the plug-in configuration in order to pick up the new URIs related to the new application.



WebSphere Application Server: base

WebSphere Application Server is the foundation for running your applications. You may use it as a stand-alone install or federate it into a cell as a node. This section will cover the WebSphere Application Server as a stand-alone system. There are many areas that could cause problems in WebSphere Application Server and the product frequently interacts with other products that could be the real cause of a problem.

The eMerge scenario uses WebSphere Application Server to host its Web site, which interacts directly with Web services and WebSphere MQ.

7.1 WebSphere Application Server

WebSphere Application Server is a complex product and includes many areas where problems can occur. You need to be ready to examine error messages in the Administrative Console, review log files, and research more information. There are many common problems that will be mentioned here, but there are also many WebSphere sources that may go into more detail about existing problems or cover other ones.

In the eMerge scenario, WebSphere Application Server can be the beginning or end point of a variety of problems. Some interactions between components include the HTTP plug-in, messaging, Web services, security, firewalls, hardware, clients, and database connection.

In this section, many of the trouble spots in WebSphere Application Server will be covered.

7.1.1 General problem determination guidelines

The following steps apply to general problem determination on WebSphere Application Server. These guidelines can be useful in both development and runtime environments.

1. An error occurred with the application server; we assume that the problem is with the application server or with the application running on it. The easiest way to figure this out is to check the log file for WebSphere:
<WebSphere_root>/logs/SystemOut.log.
2. First, check the log file and look for any error during WebSphere startup; if there is an error before the point where you see "...ready for e-business", then the error is probably with the application server configuration. If it is not an obvious error then go ahead with the log file and look for other symptoms.
3. Find the first occurrence of the error in the log file. You will either see an error message or an error exception with a stack trace.

If you get a proper error message, it may come from WebSphere, telling you that there is something wrong with the application server. The error message can also come from the enterprise application itself if an exception was caught and handled properly; in this case, the message should describe the problem, or at least give you a pointer on where to look for the message. The exceptions that are caught but not handled well are the worst problems in any application, because it is hard to find them.

If you get a Java exception with the stack trace, there are again two options: it can come from the application server or from the enterprise application. Check the stack trace and follow the classes, with the package names, and

try to identify them. If all the classes belong to the application server and you can easily identify them by the package names, then you probably have a configuration error. Try to figure out the source of the problem based on the class names; they are usually quite descriptive.

If you find any class in the stack trace that belongs to the enterprise application, then you need to check the application. You can even find the line number from the Java source where the error occurred. Open the source file and find the line with the error.

4. If the error is with the enterprise application, obviously there can be many reasons for it. At this point, you will need to look in the source or contact the developer for help.

Keep in mind that even though you found that the problem is at the level of the enterprise application, the source of the problem can still be in the application server. For example, if the JNDI name for a queue destination is misconfigured in the application server, but the resource reference in the enterprise application is correct, then the application will fail and it will point to the JNDI lookup in your application's class, even though the problem is not there. Of course, several of these errors can be easily identified from the source where the application is obviously calling for application server resources or configurations.

5. If you are having trouble finding the problem, go through the application design and try to find any component that you forgot to consider, or any reference that you may have overlooked. Keep in mind that in J2EE, there are several references pointing to other references in descriptors everywhere. It is hard to trace these chained references and configurations, so do not overlook any configuration file or descriptor.

7.1.2 Application servers

If the application server fails to start, the best place to start looking for the cause is the JVM logs. There may be a variety of things causing a server to fail to start. Problems could include security, port conflicts, etc.

The eMerge scenario application server runs on an AIX machine and serves the main Web site. Another application server runs the business process choreographer.

Things to check

1. Port conflicts can be caused by more than one source. There could be a ghost process of your server running that still holds onto the port. Check to see if there are old Java processes running on the server. If you still have a <servername>.pid file in the logs directory for your server, see if that PID number matches any running Java process. If so, kill it.

2. If there is a true port conflict, then either change the ports for your server or find the other server causing a conflict and change it or request the change. This can occur in a co-existence situation where multiple instances of WebSphere Application Servers are running.
3. Problems with security can cause a server to fail to start. If there are security errors in the JVM logs where security was working normally previously, verify that no one changed the security settings or that any changes that *were* made are correct. Security settings are located in the cell level of the config directory in security.xml. Then, investigate where your user registry is hosted. For instance, if the LDAP machine is not working properly, then the server cannot be authenticated.

Symptom: Application server fails to start with security errors

Check the JVM logs for security messages like the ones shown in Example 7-1.

Example 7-1 Security errors appear on previously working server

```
[11/12/03 14:55:34:477 CST] 118d9aec LTPAServerObj E SECJ0369E: Authentication
failed when using LTPA. The exception is
com.ibm.websphere.security.CustomRegistryException: [LDAP: error code 32 - No
Such Object]
    at com.ibm.ws.security.registry.ldap.LdapRegistryImpl.
checkPassword(LdapRegistryImpl.java:219)...
Caused by: javax.naming.NameNotFoundException: [LDAP: error code 32 - No Such
Object]; remaining name 'o=ibm,c=us'
    at com.sun.jndi.ldap.LdapCtx.mapErrorCode(LdapCtx.java:2988)
    at com.sun.jndi.ldap.LdapCtx.processReturnCode(LdapCtx.java:2909)...
```

First, check to see if the username and the password provided for the system are correct; passwords might have been changed or might have expired.

Check that no one has changed the security settings. Otherwise, investigate the user registry (LDAP) to verify that it is working correctly.

Symptom: Application server hangs

If you find that the application stops responding to new requests, you may determine the problem as follows.

1. If the request not being responded to is sent to the HTTP Server first, send it to the application server directly by indicating the application server host and the transport port. Make sure the combination of host name and port is in the Host Aliases list of the VirtualHost.

If the application server responds to the request, then the problem is on the level of the HTTP Server or plug-in; please refer to Chapter 6, “WebSphere

Application Server : plug-in” on page 123 and 4.6, “IBM HTTP Server” on page 94 for further help.

2. Do you understand the application flow of the request which is not responded to? Is there access to other resources, such as database access, in the application flow?

If so, what about the requests which do not have access to other resources? Create a simple test JSP, containing only some text, if possible. Try to invoke the JSP; did it respond?

If it did, perhaps the Web container is normal; in that case, take a look at the resources status and usage.

3. Check and monitor the status of resources with Tivoli Performance Viewer.

Determine which resources have reached their maximum capacity, such as Java heap memory (indicating a possible memory leak) and database connections. If a particular resource appears to have reached its maximum capacity, review the application code for a possible cause:

- If database connections are used and never freed, ensure that database resources are closed correctly in the application code, for example, with ResultSet, Statement and Connection.

The problem might be with the code; you can close the connections within a `finally{}` block. Please refer to <http://www.ibm.com/developerworks/websphere/> for further recommendations about application development.

- If there is a steady increase in servlet engine threads in use, review the application synchronized code blocks for possible deadlock conditions.
- If there is a steady increase in JVM heap size, review the application code for memory leak opportunities, such as static (class-level) collections, which can cause objects never to undergo garbage collection.

4. Force the problem application server to create thread dumps (or javacore).
 - a. Confirm the SOAP port in the Administrative Console: click **Servers -> Application Servers -> <server_name> -> End Points -> SOAP_Connector_Address**, and find the port, for example: 8880.
 - b. Create a file named `dumpthread.jacl`, as shown in Example 7-2. Make sure to set the process value to the server name, for example: `server1`.

Example 7-2 dumpthread.jacl

```
# Set the process value to the server name
set jvm [$AdminControl completeObjectName type=JVM,process=server1,*]
$AdminControl invoke $jvm dumpThreads
```

c. Change to the correct directory.

```
cd <WebSphere_root>/bin
```

d. Dump the threads.

- If security is not enabled:

```
wsadmin.sh -port <soap_port> -f dumpthread.jacl
```

- If security is enabled:

```
wsadmin.sh -port <soap_port> -f dumpthread.jacl -user  
<websphere_userid> -password <password>
```

This calls for `wsadmin.sh` on UNIX and `wsadmin.bat` on Windows. The `soap_port` is the port which you determine in step a on page 139, and the default port number is 8880.

You may also use following command to create threads dump on UNIX:

```
kill -3 <PID>
```

<PID> is the process ID of the hung application server.

e. Wait one or two minutes, then repeat step d.

f. Wait another one or two minutes, then repeat step d again.

g. Three `javacorePID.TIME.txt` will be created in the Working directory of the application server.

You may find the Working directory setting in the Administrative Console by clicking **Servers -> Application Server -> <server_name> -> Process Definition**. The default Working directory is the WebSphere installation root.

You can compare the three `javacore` files to determine whether the server is hung. You will find the threads information in the `javacore` file, which will allow you to find out what the server is doing.

The thread dump contains a snapshot of each thread in the process, starting in the section labeled Full thread dump. The code which is running and the resources which are accessed may give you clues.

- Look for threads with a description that contains "state:R". Such threads are active and running when the dump is forced, or the process exited.
- Look for multiple threads in the same Java application code source location. Multiple threads from the same location might indicate a deadlock condition (multiple threads waiting on a monitor) or an infinite loop, and help identify the application code with the problem.

For information about how to analyze the javacore files, please refer to IBM JVM Diagnostics Guide at:

<http://www.ibm.com/developerworks/java/jdk/diagnosis/>

7.1.3 Administrative Console

The Administrative Console is the GUI interface for WebSphere Application Server. You may run into problems using it to work with your environment.

Things to check

1. If you cannot open the Administrative Console, check that the server is running. The URL for the Administrative Console should look like this:
<http://hostname.domainname:9090/admin>
2. If security is enabled, verify that you have a correct user ID/password for the console.
3. Review the JVM logs to see if the Administrative Console is started. You should see this application started message: ApplicationMg A WSVR0221I: Application started: adminconsole.
4. If security is enabled with LTPA and the login page loops, check that you have the domain name included in the URL. For example:
<http://myMachine.raleigh.ibm.com:9080/admin> versus
<http://myMachine:9080/admin>.
5. Verify that there is no firewall between the system where you are using the Administrative Console and the machine running it.
6. Review the JVM logs and InfoCenter for more information about error messages that appear in the Administrative Console.
7. If there are save conflicts, review the conflicting documents to decide whether to overwrite.
8. If the status on servers or applications does not seem accurate, use the refresh icon to update the status.
9. If there are options missing, verify that your login has the proper permissions (if you are in the administrator role, configurator role or others). You may want to check that the install was successful and no components failed to install.
10. Verify that the Administrative Console was installed during WebSphere Application Server installation and that it started normally with the server.
11. If you are using a Netscape browser, review the InfoCenter article *Errors connecting to the Administrative Console from a Netscape browser*.
12. If you are forced to log in to the Console again, another user may have logged you out or your session may have timed out.

13. Enable a trace on the Administrative Console.

- a. Stop the server.
- b. Edit the server.xml file under the following directory:
`<websphere_root>/config/cells/<cell_name>/nodes/<node_name>/server
s/<server_name>`.
- c. Search for `startupTraceSpecification` and change the string from
`*=all=disabled` to `com.ibm.ejs.*=all=enabled`.
- d. Save the file `server.xml` with the above change and stop and restart the
server.
- e. Reproduce the problem by bringing up the console in your browser and
check `trace.log` for trace information and errors.

Symptom: Application listed as unavailable

Since trailing spaces are not allowed in the name of an application server, an application server created with a trailing space in its name appears on the list of application servers, but will be listed as “Unavailable” and it would not be possible to remove it using the Administrative Console.

You can solve this problem by finding and removing the directory named `<serverName>` in the directory
`<websphere_root>/config/cells/<nodename>/nodes/<nodemanagername>
/servers`.

Symptom: Administrative Console fails to save changes

If the save process takes more than 180 seconds, the Administrative Console fails to save changes with the following errors on the Console.

Example 7-3 Error failing to save

```
com.ibm.ws.scripting.ScriptingException:  
com.ibm.websphere.management.exception.ConfigServiceException
```

If you save the configurations using the **\$AdminConfig save** `wsadmin` command and if the save process takes more than 180 seconds, then you will see the following errors in `wsadmin.traceout` under the `websphere_root/logs` directory.

Example 7-4 wsadmin failing to save

```
com.ibm.websphere.management.exception.ConnectorException  
org.apache.soap.SOAPException: [SOAPException: faultCode=SOAP-ENV:Client;  
msg=Read timed out; targetException=java.io.InterruptedIOException: Read timed  
out]  
" follows:  
com.ibm.websphere.management.exception.ConfigServiceException
```



```
at
com.ibm.websphere.management.configservice.ConfigServiceProxy.save(ConfigServiceProxy.java:151)
  at com.ibm.ws.scripting.AdminConfigClient.save(AdminConfigClient.java:2211)
  at java.lang.reflect.Method.invoke(Native Method)
  at tcl.lang.reflect.PkgInvoker.invokeMethod(PkgInvoker.java:125)
```

To solve the above savings problem, edit the `com.ibm.ws.client.props` file under the `<WebSphere_root>/properties` directory and change the `com.ibm.CORBA.requestTimeout` value from 180 to any appropriate higher value, for example 600.

Symptom: Login page loops

- ▶ Check that you are using the fully qualified host name for the machine. Using localhost or a short name will cause the page to loop. An example: using `http://myMachine.raleigh.ibm.com:9080/admin` versus `http://myMachine:9080/admin`.
- ▶ One step in doing that is to provide the fully qualified domain name (DNS) in the URL specified in the browser. That name must include the domain specified as the LTPA domain in the configuration.
- ▶ You can also try a new browser window. Old login information from another Administrative Console or another program may interfere with the login.

If these steps do not work, try these:

1. Check to see if your WebSphere machine has a static IP address. If not, ping the machine by its fully qualified distinguished name and note the IP address.
2. Edit the hosts file on the machine to include an entry for that host and IP address.
3. Attempt to log in to the Administrative Console again. If the login is successful, the user should continue to be able to log in until the machine is assigned a different IP address.

Note: Specifying a static IP address to the machine would be the permanent solution.

7.1.4 wsadmin problems

`wsadmin` is the WebSphere Application Server scripting administration tool.

Things to check

1. `wsadmin` hangs with no action.

2. wsadmin has starting issues.
3. wsadmin commands fail to work.
4. wsadmin scripts (jacl, JavaScript or JPython) fail to work.

Symptoms: wsadmin hangs

If wsadmin is hanging, enable tracing on wsadmin and check the respective log files as listed below.

1. Using any editor, open the wsadmin.properties file located in the <WebSphere_root>\properties directory.
2. Uncomment the following line:

```
com.ibm.ws.scripting.traceString=com.ibm.*=all=enabled
```
3. Recreate the problem, then check the wsadmin.traceout file under the <WebSphere_root>\logs directory and all the logs under the <WebSphere_root>\logs\<server_wsadmin_connected_to> directory for actual problem determination.

Symptoms: Error creating SOAP connection to host...

The reasons for the above error can be any one of the following:

- ▶ The server wsadmin is attempting to connect to is not running.
- ▶ The port/hostname wsadmin is using is incorrect or the port is in use; for example, SOAP uses 8880 by default.
- ▶ wsadmin is not supposed to connect to a running server and needs to start with the option `-conntype NONE`.

7.1.5 Applications

Applications are a crucial part of your environment. If your application does not start or stops working, it is critical that you pinpoint the problem. In this section, we must assume that your application code is tested and working. It might be difficult to distinguish between an application bug and a problem with WebSphere Application Server or another component of your environment, but this section will concentrate on WebSphere Application Server related problems. With the aid of the various problem determination tools, you should be able to distinguish between an application development problem and a WebSphere Application Server configuration problem. Sometimes, it is a matter of process by elimination.

The eMerge scenario's application to run the main Web site could face problems with its application server crashing or with getting information from Web Services or messaging.

Things to check

1. If you have problems with application startup, verify that your server is started and running normally.
2. Look for obvious resource error problems in the JVM logs, such as incorrect user IDs/passwords on your datasources.
3. If security is enabled, check for Java2 security errors in the JVM logs; you may have to update your application's was.policy file.
4. If you have problems accessing your application, double-check that your application server and application are running.
5. Try accessing your application first via the internal HTTP Server. The default port for WebSphere Application Server server1 is 9080. You should know the context root for the Web module you want to access and the correct HTML, JSP or servlet name if necessary. Double-check that you are using the correct spelling and case.
6. If there is a firewall involved, the correct ports need to be open for WebSphere Application Server. See the WebSphere InfoCenter for a list of default ports that WebSphere Application Server uses.
7. If you are experiencing problems using an HTTP Server, please refer to 6.1, "Web server plug-in" on page 124.
8. It is possible that during application start, a part of the application suffered errors, but that these were non-fatal and you will not discover the problem until later. Review the JVM logs during application startup to see if there were problems.
9. If you implemented JRas tracing and logging in your application, it will be useful to determine both application errors and WebSphere Application Server problems. You can determine information on your application and, ideally, track down application errors. It could also help in determining where a configuration problem might be, depending on where the error exists relative to the application tracing. See the WebSphere InfoCenter on JRas tracing for more information about adding it to an application.
10. Review your application settings. It is possible that during install or after install, all the resources or JNDI names are not correct. If you are unfamiliar with the application, consult documentation for it and check that all your application settings are correct. Start with components related to any JVM errors messages that you see.

Symptom: Application does not start

Investigate this with the JVM logs. They will contain errors, messages and stack trace relating to whatever part of your application failed. Some common starting problems might include the following.

- ▶ The application server is not started. Check that server your application is installed on is started.
- ▶ If there are resource problems, investigate the error messages in the JVM logs. Correct information on your data source if you have missing or incorrect parameters. The user and password on the EJBs must match the data source user and password.
- ▶ There might also be problems with the database. For instance, there might be tables missing for your EJBs. Check to see if you need to do anything specific if you are using a two-phase commit. For example, in DB2, you must bind your database for use with the two-phase commit. In this case, you will see the following error. Use the SQL error listed in the JVM logs to look up database-specific information.

SQL0805N Package "NULLID.SQLLC300" was not found. SQLSTATE=51002
- ▶ If Java2 security is enabled and your application accesses protected WebSphere Application Server resources, then during startup or during runtime, you will see Java2 security errors. At the start of the errors listed in the JVM log, you will see the offending code. To correct this, you will either need to change the logic of the application (if you decide that you do not want to allow the access) or edit the application's was.policy file to allow the action. If you install a 1.2 level EAR file, a default was.policy file will be provided for you. It is only a policy file structure and you have to fill in specific permissions.

Example 7-5 Default was.policy file for an application

```
// Template policy file for enterprise application.
// Extra permissions can be added if required by the enterprise application.
//
// NOTE: Syntax errors in the policy files will cause the enterprise
// application FAIL to start.
// Extreme care should be taken when editing these policy files. It is advised
// to use the policytool provided by the JDK for editing the policy files
//      (WAS_HOME/java/jre/bin/policytool).
//
grant codeBase "file:${application}" {
};
grant codeBase "file:${jars}" {
};
grant codeBase "file:${connectorComponent}" {
};
grant codeBase "file:${webComponent}" {
};
```

```
grant codeBase "file:${ejbComponent}" {  
};
```

- ▶ If your application depends on other files, you might have to add them to the classes directory or into the EAR file of your application if your application cannot find them.
- ▶ If your application depends on JNDI names from another application, check that it has been installed and is running. See the name space section for using and viewing dumpnamespace.

Symptom: Application starts but there are never ending transactions

Never ending transaction messages are being produced due to an unsupported environment, when application code spins its own threads from an EJB or a servlet for accessing a database is not supported (per the J2EE specification). If a servlet is spinning its own threads and accessing a database then it is not submitting to the J2EE specifications. The following warning messages can be found in your application SystemOut.log file for this scenario.

Example 7-6 Unsupported database error messages

```
[11/15/03 11:53:41:711 EST] 891a0 ConnectionMan W J2CA0075W: An active  
transaction should be present while processing method allocateMCWrapper.  
[11/15/03 11:53:41:815 EST] 891a0 ConnectionMan W J2CA0075W: An active  
transaction should be present while processing method initializeForUOW.
```

These are only the warning messages alerting the programmer not to spin their own threads from within the code. Consider changing your application code to comply with the J2EE specifications.

Symptom: Application starts, but cannot access with a browser

If you are using an external Web server, review the Web server plug-in section for problems with the plug-in.

Verify that you are using the correct URL for the application. Review the URI for the Web module and any HTML, JSPs or servlets that you are trying to reach.

Review the logs for errors about the page you are trying to access. For instance, there might be a problem compiling a JSP.

Symptom: Cannot login, authorization failed...

The full error message is similar to this:

```
Authorization failed for /UNAUTHENTICATED while invoking resource
securityName:/UNAUTHENTICATED;accessId:UNAUTHENTICATED not granted any of
the required roles.
```

If security is enabled, it is likely that an unprotected servlet or JSP accessed a protected enterprise bean. Servlets and JSPs that access protected enterprise beans need to be protected so the user will enter a user ID that is passed onto the EJB. In this case, there was no login and the servlet or JSP runs as UNAUTHENTICATED. Make sure the servlet's runAs property is set to an ID that can access the enterprise bean.

Symptom: Browser errors, you are not authorized to view this page, Error 403: AuthorizationFailed, or HTTP 403 Forbidden errors

Review the logs for an error like: Authorization failed for <user_ID> while invoking resource securityName:/username;accessID:xxxxx not granted any of the required roles <roles>. Check the users assigned to the roles listed and either use a different user ID or add user IDs or groups to the role.

Symptom: NullPointerException

If you encounter a NullPointerException, you should first review your application code. Look in the logs to see where the null pointer occurs. If it does not occur in your code, but in WebSphere Application Server code, use the stack trace to see if your code is involved and possibly missing something to pass to WebSphere Application Server code.

You may want to use the debugger in WebSphere Studio Application Developer to step through code and track down a NullPointerException. For more information, review the WebSphere Studio Application Developer section on the debugger.

For null pointers that appear to be in the WebSphere Application Server code and not caused by your application code, contact IBM support.

Symptom: Attempt to start EJB module fails with "javax.naming.NameNotFoundException dataSourceName_CMP" exception

If you are using a datasource for a 2.0 EJB, verify that the data source has **Container Managed Persistence** selected. If it is not selected, select the checkbox, save and restart the application server.

If it is selected, verify that the JNDI names are correct and review the JVM logs for naming warnings or errors.

Symptom: Transaction [tran ID] has timed out after 120 seconds accessing EJB

You may see this error when a client executes a transaction on a CMP or BMP enterprise bean. The default timeout value for enterprise bean transactions is 120 seconds. The transaction is closed after this time. If your transaction needs to take longer, you can change the timeout value by using the Administrative Console and going to **Servers -> Application Servers -> server_name**. Select the Transaction Service properties page, and look at the property *Total transaction lifetime timeout*. Increase this value if necessary and save the configuration. If you think the transaction should not take that long, you may need to review your systems and other products to try and track down where a bottleneck occurs.

7.1.6 Tracing

If you have problems enabling trace to perform problem determination on another problem, there might be a few things wrong.

Things to check

1. Check your trace configuration settings for the correct trace string.
2. Verify the location of the trace file.

Symptom: Trace file created, but no trace information written

If you enable trace and recreate a problem, but the trace file does not contain trace information, check your trace configuration settings. If you configured trace on the Configuration tab, then you have to restart the server for it to take effect. The Runtime tab will take effect immediately.

Check that your trace string is correct and that you are enabling a trace using an enabled flag and not a leftover disabled flag. For example, `SASRas=all=disabled`.

Ensure that you exercise code that interacts with the code being traced.

Symptom: Trace file is not created

Look at the file name and path to the file name and verify that is where you want the trace to go. Check that you have enough file space for the trace since trace files can grow large rather quickly.

Symptom: Netscape browser fails when enabling trace

If you use a Netscape browser on AIX, it will fail when you try to enable trace.

Try one of these workarounds to enable trace:

- ▶ Disable JavaScript on the browser and continue setting trace.
- ▶ Use another browser and operating system.
- ▶ Change the trace manually in the server.xml file.

7.1.7 Clients

The JNDI naming changes in WebSphere Application Server 5 from previous releases. Your client may be having problems looking up JNDI names in WebSphere Application Server. For example, the JNDI name myCompany/bankAccount would need to be preceded with cell/nodes/<nodename>/servers/<servername>/ in order to look up the name from a client.

Things to check

1. Rerun the `launchClient` command specifying the `-CCverbose=true` option.
2. Check that all necessary classes and jars are in the classpath.

Symptom: Error: com.ibm.websphere.naming. CannotInstantiateObjectException...

The full error message is as follows:

```
Error: com.ibm.websphere.namingCannotInstantiateObjectException: Exception
occurred while attempting to get an instance of the object for the
specified reference object. [Root exception is
javax.naming.NameNotFoundException: xxxxxxxxxx]
```

This exception occurs when you perform a lookup on an object that is not installed on the host server. Check that the bean being looked up is installed on the server you access. Also verify that the JNDI names match between your application client and the JNDI names on the host server. Check that the

resource for the JNDI name is defined, installed, and the application server is started.

Symptom: Error: javax.naming.CommunicationException...

The full error message is as follows:

```
Error: javax.naming.CommunicationException: Could not obtain an initial
context due to a communication failure. Since no provider URL was
specified, either the bootstrap host and port of an existing ORB was used,
or a new ORB instance was created and initialized with the default
bootstrap host of "localhost" and the default bootstrap port of 2809. Make
sure the ORB bootstrap host and port resolve to a running name server. Root
exception is org.omg.CORBA.COMM_FAILURE: WRITE_ERROR_SEND_1 minor code:
49421050 completed: No.
```

This exception occurs when you run the **launchClient** command to a host server that does not have the Application Server started or if you specify an invalid host server name or no host server name when you run **launchClient**. The default behavior is for **launchClient** to run on the localhost which may not be correct if your host server is remote. Check the name of the host server and bootstrap port and whether the application server is running.

Symptom: Error: javax.naming.ServiceUnavailableException...

The full error message is as follows:

```
Error: javax.naming.ServiceUnavailableException: A communication failure
occurred while attempting to obtain an initial context using the provider
url: "iiop://[invalidhostname]"
```

Make sure that the host and port information are correct and that the server identified by the provider URL is a running name server. If no port number is specified, the default port number 2809 is used. Other possible causes include the network environment or workstation network configuration. The root exception is as follows:

```
org.omg.CORBA.INTERNAL: JORB0050E: In Profile.getIPAddress(),
InetAddress.getByName[invalidhostname] threw an UnknownHostException. minor
code: 4942F5B6 completed: Maybe
```

You will see this exception if you specify an invalid host server name.

Symptom: The launchClient command appears to hang and does not return to the command line when the client application has finished

If your application client needs to display the security login dialog, and System.exit is not called, the code will end because it is waiting for the Abstract

Windows Toolkit security dialog thread to exit. You should either modify your application to call `System.exit(0)` as the last statement or use the `-CCexitVM=true` parameter when you call the **LaunchClient** command.

Symptom: J2EE client request hangs and cannot recover when the requested resource is disconnected

When a J2EE client requests a Web resource from a cluster, if the member on which the requested resource resides is disconnected, the client application may hang. The application cannot be recovered within the WLM environment.

To correct the problem, launch the J2EE client with the following parameters:

```
-CCDcom.ibm.CORBA.RequestTimeout=10  
-CCDcom.ibm.websphere.wlm.Unusable.Interval=180
```

7.1.8 Resource providers

There are many types of resources for an application to use. JDBC, MQ, and other resources can be the source of problems for applications. They can also be a starting point for finding problems that actually exist in other products. This section will cover the WebSphere Application Server side of resources. If you determine that the WebSphere Application Server resource providers are configured correctly, you will need to investigate the database, queue or other outside resource that WebSphere Application Server is trying to use.

In the eMerge scenario, we specifically need to investigate MQ and JDBC resource problems. A problem with the WebSphere MQ JMS Provider or with the queue on WebSphere MQ would cause production to slow or halt.

For Enterprise-specific resources such as Extended Messaging and Work Manager, see the WebSphere Enterprise information in Chapter 8, “WebSphere Application Server Enterprise” on page 173.

7.1.9 Resource providers: JDBC providers

JDBC providers can cause problems that show up later when their data sources are used. If you see problems with an application starting or using a data source, there might be a problem with the JDBC provider. Start troubleshooting by checking the JVM logs for error messages.

Things to check

1. There may be an incorrect classpath to the JDBC provider classes. The Classpath field must include the location of the files on all machines that will have data sources from the JDBC provider. Either edit the classpath in the

Administrative Console under **Resources -> JDBC provider -> <your_provider>** or edit the WebSphere variable for it.

2. The JDBC provider files are not on the machine or are not all the files that you need. Verify that the classes exist and are not corrupt. Check that you have the right number of files to reference. For example, with Informix® Dynamic Server, you might need to reference two files for the JDBC driver.
3. The JDBC provider files are not at the correct level for either the database or the WebSphere Application Server.
4. The files are there and the path is correct, but WebSphere Application Server cannot read them. Check the permissions on the file and that the db2profile is sourced.
5. You might be using the wrong JDBC provider for your type of database. There are many DB2 providers to choose from and you need to select the correct one for your version and type of data source.
6. The scope of the JDBC provider might prevent you from using any subsequent data sources if you choose a scope that is too narrow. For instance, if you select a JDBC provider at the server level, then only a specific server will be able to use that JDBC provider.

Enabling database trace

Use the following trace strings for the different databases:

- ▶ com.ibm.ws.database.logwriter Trace string for databases that use the GenericDataStoreHelper. You can also use this trace string for unsupported databases.
- ▶ com.ibm.ws.db2.logwriter Trace string for DB2 databases.
- ▶ com.ibm.ws.oracle.logwriter Trace string for Oracle databases.
- ▶ com.ibm.ws.cloudscape.logwriter Trace string for Cloudscape™ databases.
- ▶ com.ibm.ws.informix.logwriter Trace string for Informix databases.
- ▶ com.ibm.ws.sqlserver.logwriter Trace string for Microsoft SQL Server databases.
- ▶ com.ibm.ws.sybase.logwriter Trace string for Sybase databases.

Your back-end database needs to support JDBC tracing, otherwise you will not get any trace data with the trace strings. The following databases offer JDBC tracing at this time:

- ▶ DB2
- ▶ Oracle
- ▶ SQL Server

To set the level of trace for DB2 Universal database, you have to set the following custom properties on the datasource. The InfoCenter lists the properties needed to do this. The DB2 Universal JDBC driver provider Custom properties for DB2 are as follows.

- ▶ **traceLevel:** possible traceLevel values are as follows.
 - TRACE_NONE = 0
 - TRACE_CONNECTION_CALLS = 1
 - TRACE_STATEMENT_CALLS = 2
 - TRACE_RESULT_SET_CALLS = 4
 - TRACE_DRIVER_CONFIGURATION = 16
 - TRACE_CONNECTS = 32
 - TRACE_DRDA_FLOWS = 64
 - TRACE_RESULT_SET_META_DATA = 128
 - TRACE_PARAMETER_META_DATA = 256
 - TRACE_DIAGNOSTICS = 512
 - TRACE_SQLJ = 1024
 - TRACE_ALL = -1

Note: This trace level provides real data that sets to the PreparedStatement or gets from the ResultSet object.

- ▶ **traceFile:** use this property to integrate the DB2 trace with the WebSphere Application Server trace. If you do not set the value, traces are integrated. Otherwise, DB2 traces are directed to the desired file. You can dynamically enable or disable trace. You can run an application and turn on the DB2 trace if there is a problem. Use the runtime trace enablement provided with the Application Server by specifying a trace string of `com.ibm.ws.db2.logwriter=all=enabled`.

Symptom: Cannot use JDBC Provider or datasource

Check the JVM log for classpath related errors as shown in Example 7-7.

Example 7-7 Incorrect classpath

```
[11/14/03 15:54:20:224 EST] 447a4cd7 DataSourceCon E DSRA8040I: Failed to
connect to the DataSource. Encountered : java.lang.ClassNotFoundException:
DSRA8000E: No jar or zip files found in C:/SQLLI/java12/db2java.zip
    at com.ibm.ws.rsadapter.DSConfigurationHelper.
loadDataSourceClass(DSConfigurationHelper.java:1030)
```

Correct your classpath to point to the right files.

Symptom: java.lang.AbstractMethodError in the JVM log

On Oracle, you may see a `java.lang.AbstractMethodError` error if your driver file is at the wrong level. Use the correct level for Oracle and WebSphere Application Server.

Symptom: java.lang.UnsatisfiedLinkError in the JVM log

In the JVM log, you might see something similar to Example 7-8.

Example 7-8 java.lang.UnsatisfiedLinkError

```
1/24/03 12:02:55:763 MST] 6f2067 FreePool E J2CA0046E: Method
createManagedConnctionWithMCWrapper caught an exception during creation of the
ManagedConnection for resource jdbc/ARCHDB1Datasource, throwing
ResourceAllocationException. Original exception:
com.ibm.ws.exception.WsException: DSRA0080E: An exception was received by the
Data Store Adapter. See original exception message:
[IBM][JDBC Driver] CLI0600E Invalid connection handle or connection is closed.
SQLSTATE=S1000.
```

The problem is that your db2profile is not sourced.

Additional information

You can find more information about problem determination with WebSphere and DB2 at:

<http://www-106.ibm.com/developerworks/db2/library/techarticle/0204vangennip/0204vangennip1.html>

7.1.10 Resource providers: data sources

Data sources have many fields where incorrect information will cause errors in other activities. For instance, after you configure a data source, install an application and assign data sources to EJBs, you might have errors while starting the application because it cannot reach the database. The JVM logs include errors messages that point to the cause of the problem.

Tip: Using Test Connection in the Administrative Console on your data source is useful for doing what its name implies: testing the connection to the database. The machine that runs the Administrative Console must also have a client connection or have the database be local for it to run a test. In a base install, this will probably be the case already, but in cell with a Deployment Manager, it might not.

Errors with Test Connection will appear in the Administrative Console. Additional error messages will appear in the JVM logs.

You might see the following errors using Test Connection, while starting an application that uses the data source or when application code looks up a data source to use it.

Things to check

1. User ID and passwords are correct for the database you are using.
2. Are there missing required custom properties for your database? A DB2 data source for a 1.1 EJB does not require custom properties, but an Informix data source needs four custom properties. The data source's Custom Properties page displays whether or not the property is required for the data source. If all of these fields are correct, consult with the database administrator about whether you need to set additional customer properties.

For example, running a test connection for Informix, you might see this message in the Administrative Console.

Example 7-9 Administrative Console error message

```
Test Connection failed for datasource InsuranceDB on server server1 at node
ka6brmy1 with the following exception: java.lang.Exception:
java.sql.SQLException: com.informix.asf.IfXASFException: Attempt to connect to
database server (hpws06) failed.. View JVM logs for further details
```

Checking the JVM logs in this case does not yield additional information. The first thing to check would likely be data source configuration parameters.

Example 7-10 JVM log error message

```
[11/14/03 15:33:45:368 EST] 39764cba DataSourceCon E DSRA8040I: Failed to
connect to the DataSource. Encountered : java.lang.Exception:
java.sql.SQLException: com.informix.asf.IfXASFException: Attempt to connect to
database server (hpws06) failed.
    at com.ibm.ws.rsadapter.DSConfigurationHelper.
testConnectionForGUI(DSConfigurationHelper.java:1786)
    at java.lang.reflect.Method.invoke(Native Method)...
```

Checking the Informix datasource reveals that the `ixlFXHOST` was set to `localhost`, but the Informix database was remote. After updating this parameter, Test Connection worked normally.

3. JDBC provider may be configured incorrectly. See 7.1.8, “Resource providers” on page 152 for more information about this.
4. You may have more than one datasource configured with the same JNDI name. Use `dumpnamespace` to see if your datasource is listed, or manually check your datasource JNDI names. There is a fix available for WebSphere Application Server 5.0.2 that will throw an error if you try to set up two data sources with the same JNDI name. See PQ76254 on the WebSphere Application support and download page for more information.

Symptom: Deadlock

You should see an error in the JVM logs stating that a deadlock was detected. Review your access intent and isolation settings for your EJBs. Review your database logs and statistics to gather more information about when the deadlock occurred. Perhaps you can enable trace to locate when the deadlock is occurring and for what part of your application. Review application code for potential deadlock causing areas.

Another option to resolve deadlock is to use the WebSphere Application Server Enterprise feature called application profiling.

Symptom: IllegalConnectionUseException

If your application uses `WAS40DataSources`, this error is caused when a connection is used on more than one thread. This could happen with servlets or BMPs. Review your code and look for connections being shared. If you store a connection in an instance variable in a servlet, the connection would be used on multiple threads at the same time.

Symptom: ConnectionWaitTimeoutException

Review your log for the frequency of this error and your CPU usage when you receive it. Determine which datasource this error is originating from and review its connection pool settings.

If your maximum number of connections is too low, there will be a high demand for connections. You may see symptoms of low CPU utilization while receiving these methods. Try setting the maximum number of connections higher.

If your connection wait time is too low then connections will time out before connections are available. If you are using the maximum number of connections and receiving this error regularly, try changing your connection wait timeout value.

If connections are not being closed or it takes a long time for them to return to the pool, review your code to make sure connections are closed and returned. You may see this occur after all the connections in a pool are used and you may get this error frequently after that.

Your server or back-end system may not be able to handle the load. Review your hardware and connection pool settings.

7.1.11 Resource providers: WebSphere MQ JMS Provider resources

There are three types of JMS resources and it is important to select the correct type for the kind of JMS provider you will use. There are provider resources for the internal WebSphere Application Server JMS, using WebSphere MQ and another party JMS provider.

- ▶ WebSphere JMS Provider. Use this for WebSphere Application Server embedded messaging.
- ▶ WebSphere MQ JMS Provider. Use this for external WebSphere MQ messaging.
- ▶ Generic JMS providers. Use this for other types of JMS providers.

These are how WebSphere MQ objects are defined within WebSphere Application Server, and their JNDI names assigned. There are four types of resources; WebSphere MQ Queue Connection Factories, WebSphere MQ Topic Connection Factories, WebSphere MQ Queue Destinations and WebSphere MQ Topic Destinations. Common problems include incorrect security permissions or name not found exceptions.

Things to check

1. Are the resources defined in the correct scope?
2. Is the user account entered for the resource as a member of the mqm group?
3. Are the Classpath and Native Library Path entered correctly for the provider? This uses the variable MQJMS_LIB_ROOT which is not set by default when no internal messaging is installed.
4. Has internal messaging or the Java Messaging component of WebSphere MQ been installed?
5. Check that the server has been restarted since the resources were defined. Resources are only bound when the server is started.
6. Verify, from the logs, that the resources were bound and that they were bound with the correct name.

See 7.1.17, “Naming” on page 168 for more on naming exceptions.

7.1.12 Embedded Messaging

Embedded Messaging is a JMS provider that comes with WebSphere Application Server. You can use it for point-to-point and publish/subscribe messaging.

You may have problems with messaging that also relate to your application's messaging beans or with the JMS resources defined for your application.

Things to check

1. Check that Embedded Messaging is installed.
2. If installed, check for a correct installation.
3. Is Embedded Messaging configured correctly?
4. Is Embedded Messaging started correctly?
5. Is the Listener process listening on TCP/IP port (the default is 5558)?
6. Are JMS resources bound correctly?

Symptom: Embedded Messaging does not work due to install problems

The Embedded Messaging feature is installed by default. If Embedded Messaging has been installed, a `websphere_root\mq_install.log` file will exist; if this file does not exist, you should suspect that Embedded Messaging was not installed. There should also be a configuration file under `websphere_root` with the name `createmq.<nodeName>_<serverName>.log`. Also check for `log.txt` under the `websphere_root\logs` directory for entries indicating that Embedded Messaging was installed.

Inspect the `websphere_root\mq_install.log` file to see if Embedded Messaging installed cleanly or not.

A clean install will be indicated by following text in `mq_install.log` file

Example 7-11 Successful install

```
15 Nov 2003 10:55:08 <<Function Successful>> return code  
WASM_ERROR_SUCCESS (0)
```

An unclear install will be indicated by the following text in the `mq_install.log` file.

Example 7-12 Unsuccessful install

```
15 Nov 2003 11:45:56 << FUNCTION FAILED>> return code  
WASM_ERROR_PREREQ_MQSERIES_SERVER_CSD (8)
```

Inspect the `websphere_root\createMQ.<nodeName>_<serveName>.log` file to see whether or not the configuration of Embedded Messaging completed correctly. A non-zero return code does not always indicate a problem.

Example 7-13 Configuration log

```
Issuing: webspdeletebroker WAS_vaughton3e_server1 -w BIP8013E:  
Component does not exist.  
A component may only be used if it has first been created.  
No user action required. rc=13
```

This is expected, belt-n-braces coding to ensure that an existing broker of the same name is recreated.

In the following case, the Embedded Messaging queue manager already exists.

Example 7-14 Configuration log where the queue manager already exists

```
Issuing: crtmqm -u SYSTEM.DEAD.LETTER.QUEUE -cc amqwasoa -om amqwasoa -os 32  
-od port=0,Exit Reason=16 -If 512 -Is 60 WAS_vaughton3e_server1  
AMQ8110: WebSphere MQ queue manager already exists. rc=8
```

Resource not found messages are acceptable if the queue manager already exists since these resources will have already been deleted.

Example 7-15 Missing channel in MQ

```
AMQ8227: Channel SYSTEM.DEF.SVRCONN not found.
```

Symptom: Queue manager or broker does not start

Check the server `SystemOut.log` file and look for the starting queue manager and broker messages, as shown below.

Example 7-16 Queue manager and broker starting normally

```
[11/15/03 13:21:04:344 EST] 68d5baa6 JMSEmbeddedPr A MSGS0050I: Starting the  
Queue Manager  
[11/15/03 13:21:44:912 EST] 68d5baa6 JMSEmbeddedPr A MSGS0051I: Queue Manager  
open for business  
[11/15/03 13:22:01:254 EST] 68d5baa6 JMSEmbeddedPr A MSGS0052I: Starting the  
Broker
```

[11/15/03 13:22:14:876 EST] 68d5baa6 JMSEmbeddedPr A MSGS0053I: Broker open for business

If these messages are missing then check that the JMS Server initialState is set to Started. Any exceptions that occurred while starting the queue manager or broker indicate problems.

Symptom: JMS Server fails to start

The WebSphere MQ queue manager runs as a set of separate processes. One of these processes is known as the listener (runmqslr) process. The listener listens on a TCP/IP port; the default is 5558. If the listener is unable to bind to this port, the JMS Server will fail. The broker has a direct connection to the TCP/IP port; the default is 5559. If the broker is unable to bind to this port, the JMS Server will fail. JMS security uses port 5557; if JMS security is unable to bind to this port, the JMS Server will fail. If there are multiple base application servers, each with a JMS Server or multiple JMS Servers, ensure that all ports are unique.

Symptom: JMS resources are not found

JMS resources are bound during startup. If, for some reason, these definitions become corrupt (for instance, following the manual editing of resource xml files), they may not be correctly bound at start up. Correctly bound connection factories, queues and topics will appear in the SystemOut.log file as:

Example 7-17 JMS resource being bound on startup

```
[11/15/03 13:52:15:156 EST] 68375 9b3 ResourceMgrIm I WSVR0049I: Binding  
PlantsByWebSphereConnectionFactory as plantsby/InvQCF
```

After server startup, if no bind entry appears in SystemOut.log for a connection factory, queue or topic, or an exception or error appears associated with the resource, then there may be a problem with the resource definition in the resources.xml file.

Check that the queue has been defined to the JMS Server. The error message in the SystemOut.log file will look like following.

Example 7-18 Queue has not been defined

```
javax.jms.InvalidDestinationException: MQJMS2008: failed to open MQ queue  
linked exception com.ibm.mq.MQException: MQJE001: Completion Code 2,  
Reason 2085
```

Message prefixes:

Messages and exceptions likely to be encountered using Embedded Messaging may come from the following:

MSGSnJMS Server

WMSGn..... WAS messaging (com.ibm.ejs.jms.*)

J2CAn.....Java Connectors (j2c)

AMQn..... WebSphere MQ

MQJMSnJMS client

BIPnBroker

7.1.13 Security

WebSphere Application Server security protects your WebSphere environment. It can protect your Administrative Console, your servers and your applications. It will enforce authenticated and authorized use to resources. Security can be most difficult to deal with during configuration. There are problems that could crop up during runtime such as communication failures between WebSphere Application Server and the user registry.

The eMerge scenario needs security enabled to log in customers and employees with a user name and password who use the site to review and apply for policies.

Things to check

1. Review the JVM logs for stack traces related to security and which security initialized. Use the InfoCenter to find more information on specific errors messages.

Example 7-19 Security initializing normally with local OS user registry

```
SASRas      A JSAS0001I: Security configuration initialized.
SASRas      A JSAS0002I: Authentication protocol: CSIV2/IBM
SASRas      A JSAS0003I: Authentication mechanism: SWAM
SASRas      A JSAS0004I: Principal name: MYHOSTNAME/aServerID
SASRas      A JSAS0005I: SecurityCurrent registered.
SASRas      A JSAS0006I: Security connection interceptor initialized.
SASRas      A JSAS0007I: Client request interceptor registered.
SASRas      A JSAS0008I: Server request interceptor registered.
SASRas      A JSAS0009I: IOR interceptor registered.
NameServerImp I NMSV0720I: Do Security service listener registration.
```

```
SecurityCompo A SECJ0242A: Security service is starting
UserRegistryI A SECJ0136I: Custom Registry:com.ibm.ws.security.registry.nt.
NTLocalDomainRegistryImpl has been initialized
SecurityCompo A SECJ0202A: Admin application initialized successfully
SecurityCompo A SECJ0203A: Naming application initialized successfully
SecurityCompo A SECJ0204A: Rolebased authorizer initialized successfully
SecurityCompo A SECJ0205A: Security Admin mBean registered successfully
SecurityCompo A SECJ0243A: Security service started successfully
SecurityCompo A SECJ0210A: Security enabled true
```

2. Check that your user registry is running normally. For instance, if you have an LDAP user registry, check that the machine is available and that LDAP is running.
3. Review your security related settings: user registry (LDAP, custom, local), LTPA, SSO, SSL, Java2, global security.
4. Be sure that you are using the correct user name and password that goes with the user registry enabled in the Administrative Console. Verify that your username exists in the registry and that you are using the correct format for your ID.
5. Does the failing task work when security is disabled? If it does, it is likely not a security problem unless you have an application configuration problem where security information is relied upon.
6. Try to determine whether there is an authentication or an authorization problem, for instance with passwords or if an authenticated user does not have permission to access a resource.
7. If you are using LTPA, check that your current LTPA keys are distributed to all the servers involved.
8. If you configured Single Sign-n (SSO), verify that you have entered the correct realm (for instance, your scope may be too narrow, as in `raleigh.ibm.com@` versus `ibm.com`).
9. Review the JVM logs for errors related to the Secure Socket Layer (SSL). See 7.1.14, “Security: SSL” on page 165 for related problems.

Example 7-20 SSL related error

```
[12/3/03 11:01:20:687 CST] 7063dfa9 LTPAServerObj E SECJ0371E: Validation of
the token failed because the token expired...
```

10. Review the JVM logs for errors related to Java 2 security. See 7.1.15, “Security: Java 2 security” on page 167 for related problems.

Example 7-21 Java 2 security related error

[12/1/07 14:47:26:317 EST] 4e2e4c77 SecurityManag W SECJ0314W: Current Java 2 Security policy reported a potential violation of Java 2 Security Permission. Please refer to Problem Determination Guide for further information...

11. To enable security trace, use
SASRas=all=enabled:com.ibm.ws.security.*=all=enabled.
12. For information about a variety of specific security error messages, see the WebSphere InfoCenter article, *Security components troubleshooting tips*.

Symptom: Authentication fails

If you suddenly have authentication problems, first verify that your security settings were not changed to something incorrect. Then check that your user registry is still running normally. For example, LDAP communication problems can manifest themselves in error messages such as this one:

Example 7-22 Security errors appear on previously working server

```
[11/12/03 14:55:34:477 CST] 118d9aec LTPAServerObj E SECJ0369E: Authentication failed when using LTPA. The exception is com.ibm.websphere.security.CustomRegistryException: [LDAP: error code 32 - No Such Object]
    at com.ibm.ws.security.registry.ldap.LdapRegistryImpl.checkPassword(LdapRegistryImpl.java:219)...
Caused by: javax.naming.NameNotFoundException: [LDAP: error code 32 - No Such Object]; remaining name 'o=ibm,c=us'
    at com.sun.jndi.ldap.LdapCtx.mapErrorCode(LdapCtx.java:2988)
    at com.sun.jndi.ldap.LdapCtx.processReturnCode(LdapCtx.java:2909)...
```

Symptom: Not prompted for login with application

Is security enabled? Have all the servers been restarted to enable security? Check to see what kind of users you have assigned to your application roles. If you have Everyone assigned to roles, then security lets everyone in and does not prompt for a login. Also review your Web module web.xml file for the type of login authentication method. Check that you have one assigned and that it is supported by WebSphere Application Server.

Symptom: Custom login page loops instead of logging in

Check that you are using a fully qualified hostname, for instance, `http://myMachine.raleigh.ibm.com:9080/LGIInsurance` versus `http://myMachine:9080/LIFInsurance`. Try a new browser since previous logins to other applications that did not log out could interfere with the login process.

Symptom: SECJ0237E: One or more vital LTPAServerObject configuration attributes are null or not available

If LTPA is your authentication mechanism, but the LTPA keys have not been generated then you will see this error. LTPA keys are used for encrypting the LTPA token.

To resolve this problem:

1. Select **System Administration -> Console users -> LTPA**.
2. Enter a password, which can be anything.
3. Enter the same password in the Confirm Password field.
4. Click **Apply**.
5. Click **Generate Keys**.
6. Save the configuration.

7.1.14 Security: SSL

Like WebSphere Application Server security, SSL can be difficult to configure correctly between components. Once it is working, there are a few things that can still interfere during runtime.

Things to check

1. Your certificate could expire. When this happens, you will see SSL errors and the components talking to each other over SSL will no longer work correctly. To fix this, issue new certificates.
2. A false indication of expired certificates could take place if the date or time changes on the machines involved in SSL communication. Check that the times and dates between systems are synchronized.

Symptom: Security fails, problems with SSL token

Verify that your certificate has not expired. Review the times and dates on your machines to make sure they are accurate.

Example 7-23 SSL token error

```
[12/3/03 11:01:20:687 CST] 7063dfa9 LTPAServerObj E SECJ0371E: Validation of
the token failed because the token expired. If the token is coming from a
different WebSphere node or cell make sure the date and time (including the
time zone) are synchronized between all the nodes and cell(s) involved. One can
consider increasing the token timeout value if necessary.
```

```
[12/3/03 11:01:21:393 CST] 7063dfa9 JaasLoginHelp A SECJ4034I: Token Login
failed. If the failure is due to an expiring token, verify the system date
```

and time of the WebSphere nodes are synchronized or consider increasing the token timeout value. Authentication mechanism system.LTPA and exception is com.ibm.websphere.security.auth.WSLoginFailedException: LTPAServerObject: Token expired

```
    at com.ibm.ws.security.ltpa.LTPAServerObject.validate
(LTPAServerObject.java:461)
    at com.ibm.ws.security.server.lm.ltpaLoginModule.login
(ltpaLoginModule.java:353)
    at com.ibm.ws.security.common.auth.module.proxy.WSLoginModuleProxy.
Login(WSLoginModuleProxy.java:119)...
```

To solve the above problem, the system time on the machine was changed to the correct time.

Symptom: SSL problems, SSL handshake fails

If your certificates are not correct, you may see an SSLHandshakeException. Verify that you have correct certificates for all of the components talking to each other over SSL.

Example 7-24 SSL handshake fails

```
Root exception is org.omg.CORBA.COMM_FAILURE:
CAUGHT_EXCEPTION_WHILE_CONFIGURING_SSL_CLIENT_SOCKET: JSSL0080E:
javax.net.ssl.SSLHandshakeException - The client and server could not negotiate
the desired level of security. Reason: unknown certificate minor code:
49421070 completed: No
    at com.ibm.ws.security.orbssl.WSSSLClientSocketFactoryImpl.createSSLSocket
(Unknown Source)
    at com.ibm.ws.orbimpl.transport.WSSSLTransportConnection.createSocket
(Unknown Source)
```

The SSL handshake errors can be tracked back to several different causes.

1. First, you have to eliminate any possible configuration errors.
2. Check the certificates and make sure that they are not expired, not revoked and that they are valid.
3. Even though you have the correct certificates, make sure that your systems support the right encryption algorithms and key sizes so they can choose ones that are common to both parties.
4. SSL is using different ports for communication, so make sure the parties can communicate over the right channels and ports.

7.1.15 Security: Java 2 security

Java 2 security protects system resources such as file I/O, sockets, and properties. Access permission is defined by policy files. Java 2 security is enabled by default when you enable global security.

Things to check

1. Look for `AccessControlExceptions` in the JVM log. This indicates that something violated the Java 2 security policy. The top of the stack should start with the code violating the policy.
2. Review the `was.policy` file for your application. Make sure you have correct permissions for your code. Review the InfoCenter for more information on using Java 2 security.
3. Review your code if it is accessing protected resources but should not be.

Symptom: Application fails to start, Java 2 security errors in log

Review the JVM logs for Java 2 security errors. They are likely to be long stacks of Java 2 security policy exceptions and stack exceptions. At the top, the code that caused the original violation is listed.

Example 7-25 Java 2 security error stack trace from the JVM log

```
[12/1/07 14:47:26:317 EST] 4e2e4c77 SecurityManag W SECJ0314W: Current Java 2 Security policy reported a potential violation of Java 2 Security Permission. Please refer to Problem Determination Guide for further information.
```

Permission:

```
C:\WebSphere\AppServer : access denied (java.io.FilePermission C:\WebSphere\AppServer read)
```

Code:

```
com.ibm.wspi.bookstore.logging.BookStoreHandler in {file:/C:/WebSphere/AppServer/installedApps/ka6brmy1/Griffin.ear/logging.jar}
```

Stack Trace:

```
java.security.AccessControlException: access denied (java.io.FilePermission C:\WebSphere\AppServer read)
    at java.security.AccessControlContext.checkPermission
    (AccessControlContext.java:286)
    at java.security.AccessController.checkPermission(
    AccessController.java:413)
    at java.lang.SecurityManager.checkPermission(SecurityManager.java:565)
    at com.ibm.ws.security.core.SecurityManager.checkPermission
    (SecurityManager.java:168)
```

```
at java.lang.SecurityManager.checkRead(SecurityManager.java:910)
at java.io.File.exists(File.java:560)
at com.ibm.wspi.bookstore.logging.BookStoreHandler.openDevice
(BookStoreHandler.java:697)...
```

In this stack, there is a read violation error of protected WebSphere Application Server code. To solve this problem, addition permissions could be given to the application in its was.policy file or the code could be modified to not read the protected code.

7.1.16 Console users and groups

Assigning users to Administrative Console roles can help prevent problems by authorizing different people to do different things. For instance, in a moderator role, a user can view the status and settings in the Administrative Console, but not make changes or stop and start servers.

When you assign users or groups, you must already know the names that you want to use. You cannot look up names to assign them. If you enter a user or group not in the user registry, then you will get an error message. Console users and groups are stored in the admin-authz.xml file under the following directory: `websphere_root/config/cells/<cell_name>/admin-authz.xml`.

Things to check

1. Check that you have the correct spelling for the user or group in question.
2. You must have a user registry configured in order to assign users and groups to roles.
3. See 7.1.13, “Security” on page 162 for more things to check.

7.1.17 Naming

You may have problems with the naming service. The naming service can help you find JNDI names or note the absence of names that you need. The `dumpNameSpace` tool lists the name space for you.

Things to check

1. Review the JVM logs for what code is missing what kind of name.
2. Use `dumpNameSpace` and review it for the names that you need; see 4.3.11, “DumpNameSpace” on page 65 for more information on this command.
3. Check that the JNDI name that the application is looking for is correct.

4. If a client is having JNDI problems, check that all necessary servers and applications are running.
5. Ensure that the client is performing the name lookup on the correct server. Each application server has its own naming URI and unless the client is looking up the fully qualified name (which is very unusual) then it is vital that the correct URI be used. By default, this is `iiop://localhost:2809`, but in the instance where the server is part of a WebSphere Application Server cluster, the nodeagent uses this URI and the client must be set to use a non-default port. This can be determined by examining the value of the `BOOTSTRAP_ADDRESS` endpoint of the server in question.
6. It is possible to use the IBM Universal Test Client (see “Universal Test Client” on page 104) to browse the namespace in a tree-like structure. This should allow you to determine if the name you are searching for has been successfully entered into the namespace.
7. Enable trace with `com.ibm.ws.naming.*=all=enabled:`
`com.ibm.websphere.naming.*=all=enabled.`

Symptom: Name not found

If your name is not found, review the JVM logs for what code is missing the JNDI name.

1. Review the JNDI name settings.
2. If the JNDI name depends on another application or server running, check that those processes are running normally.
3. Run a `dumpNameSpace` to check for the presence of the JNDI name.

7.1.18 Java Virtual Machine: Memory

The Java Virtual Machine (JVM) is an interpretive computing engine responsible for executing the byte codes in a compiled Java program. The JVM translates the Java byte codes into the native instructions of the host machine. WebSphere Application Server, being a Java process, requires a JVM in order to run and to support the enterprise applications running on it. Common problems that can occur in JVM are memory leak issues.

Things to check

1. The JVM may run out of Java heap space to allocate a Java object.
2. The JVM may not be able to allocate the native memory that it needs to allocate for an object.
3. The JVM is not able to allocate a resource, such as a file handle to instantiate an object.

4. A `java.lang.OutOfMemoryError` is often but not always indicative of a memory leak. A memory leak occurs in Java when an object is never de-referenced in the JVM. The first thing that needs to be done is to determine if the `java.lang.OutOfMemoryError` is occurring due to the Java heap, not having enough space or the native memory running out.

Symptoms

Enable the verboseGC output for the JVM. This will log garbage collection events to the `StandErr.log`. The following example shows the GC event entries on the Solaris platform.

Example 7-26 GC event entries

-
1. [GC 139K->97K(253440K), 0.0037673 secs]
 2. [Full GC 104851K->11414K(253504K), 0.7971369 secs]
-

The first event is an incremental GC event. These are not full GC events; they are quick events that free a small amount of memory and complete in a short period of time. The numbers in these events show the amount of heap memory in use before the GC event (139K); the amount of GC in use after the GC event (97K); the size of the heap itself in parentheses (253440K); and the amount of time that the GC event took (0.0037673 secs).

The second event is a full GC event indicating the garbage collection of the entire heap. These are longer events, in terms of time, and normally free significantly more memory than an incremental GC event. The numbers in these events show the amount of heap memory in use before the GC event and the amount of GC in use after the GC event; the size of the heap itself is in parentheses. Finally, the amount of time that the GC event took is recorded is shown. In the full GC event above, there were 11414K of memory used by the heap before the GC event; after the GC had completed, there were 253504K in the heap and it took 0.7971369 seconds to complete the GC.

The point to be noted is the present size of the heap. The minimum size of the heap is the `-Xms` value set on the Java command line. The maximum size of the heap is the `-Xmx` value set on the Java command line.

What you should look for in the GC events is whether the amount in use space after the full GC event is approaching the maximum heap size. If this is not the case, then the `java.lang.OutOfMemoryError` is likely not due to the Java heap.

Also, you should note how long the GC events are taking. As a general rule, the percentage of time taken by the GC should be less than 20% of the total time. This can be determined using the GC events. A JVM that is having objects leaked will see the percentage of time taken by GC grow; this means that the time taken

by each individual GC event will likely be larger than earlier in the StandardErr.log file, the time between GC events will be less, and the amount of memory freed will be minimal.

7.1.19 HTTP Sessions

An HTTP Session is a series of requests to a servlet that originates from the same browser. Sessions allows applications running in the Web container to track individual users.

In our scenario, sessions are used to distinguish users by session IDs, preventing situations where more than one user is using our application at the same time to serve the user with information that was generated based on another users's inquiry.

In WebSphere Application Server, HTTP Sessions are managed by the Session Manager running on WebSphere Application Server.

For a detailed description of HTTP Sessions and problem determination steps with typical symptoms, refer to 9.1.5, "HTTP session management" on page 236.

7.1.20 Performance

The scope of this document does not include performance analysis. Errors or an incorrect configuration of WebSphere Application Server will certainly slow or halt performance. If performance is low and there are error messages in the logs, then you should investigate the component that is causing the problem. To increase performance on a system that is working correctly, please refer to the IBM Redbook *IBM WebSphere V5.0 Performance, Scalability, and High Availability: WebSphere Handbook Series*, SG24-6192.



WebSphere Application Server Enterprise

WebSphere Application Server Enterprise provides many helpful and interesting features. Of course, these features can add to the complexity of problem determination by adding layers or simply more elements to your runtime. As with WebSphere Application Server and WebSphere Deployment Manager, there are many points where problems may be found, frequently between the WebSphere Application Server Enterprise product and other products with which it interacts during configuration and runtime.

The eMerge scenario specifically uses the business choreographer to do business. We will also cover the many other features present in WebSphere Application Server Enterprise.

8.1 WebSphere Application Server Enterprise

This chapter provides detailed problem determination guides for WebSphere Application Server V5.0 Enterprise. The base WebSphere Application Server related topics are covered in the previous chapter, here we are going to focus on the Programming Model Extensions (PME), especially the Process Choreographer.

8.1.1 Process Choreographer

The Process Choreographer allows an application to choreograph all kinds of business processes. A process can include human interaction, Web services, business transactions and many other interactions.

Things to check

1. Review the logs. All messages that belong to Process Choreographer are prefixed with BPE. The message format is BPE<component><Number><TypeCode>. Process Choreographer messages can be found in the SystemOut.log, the SystemError.log, and traces. To view Process Choreographer messages, check the activity.log file by using the WebSphere Log Analyzer.
2. To enable trace for process related events, use `com.ibm.bpe.*=all=enabled`.
3. Review process-related audit trail information. When a process instance is executed and audit trailing is enabled, Process Choreographer writes information about each significant event into an audit log, which is located in the corresponding database table. Process Choreographer provides a plug-in for the audit trail database to be used. In addition, you can clean up the audit log table according to your needs by using the cleanup utility.

The **BPEAuditLogDelete** utility (refer to the tools section to know more about this utility) describes how to start the utility and lists the options that you can use.

Troubleshooting the business process container

Here are some things to check if you have problems getting the business process container to work:

1. Tables or views cannot be found in the database

When configuring the authentication alias for the data source, you must specify the same user ID that was used to create the database (or to run the scripts to create it).

2. You get a database error when installing an enterprise application that contains a process
When an enterprise application is installed, process templates are written into the Process Choreographer database. Make sure that the database system used by the business process container is running and accessible.
3. Cannot invoke Cloudscape tools
Make sure that you have set up the Java environment, and have included the necessary JAR files in the classpath environment variable.
4. Problems using national characters
Make sure that your database was created with support for Unicode character sets.
5. Problem creating the queues using WebSphere MQ, .dll not found
Add the WebSphere MQ Java lib directory to your path environment variable.
6. Problem on AIX connecting to the queue manager
Edit the file /var/mqm/mqs.ini, and add the following property to the definition for your queue manager: IPCCBaseAddress=12.
7. Deadlocks in DB2
To avoid deadlocks, be sure that the DB2 isolation level is set to `read stability`. If necessary, enter the command `db2set DB2_RR_TO_RS=YES` and restart the DB2 instance to activate the change.

Symptom: BPEA0010E: Unexpected exception during execution

Error BPEA0010E can occur in the following two situations:

- ▶ When a user tries to start a process in the Web client, but the queue manager is not started. In this case, the user receives a message similar to this one:
BPEA0010E: Unexpected exception during execution
To find out whether the QueueManager is running properly, perform the following steps.
 - a. Open the WebSphere systemout.log file.
 - b. Find the following error message in the log file:

```
javax.jms.JMSEException:  
MQJMS2005: failed to create MQQueueManager for '..._<servername>'
```
 - c. If this error is found in the log file, start the Queue Manager and verify that it is running properly.

- d. If the Queue manager was down before step 3 and is running after step 3, try to start the process in the Web client again.
- ▶ When a user tries to start a process in the Web client, but the database access does not work. In this case, the user receives a message similar to this one:

BPEA0010E: Unexpected exception during execution

Find out whether the database and the access to it from WebSphere are working.

Example 8-1 Database is not running

if the database is DB2 and your operating system is Windows, the password for the user ID responsible for accessing DB2, should be specified in 3 different places:

The Windows User Registry for the db2admin user ID

The "Log On" tab for the properties of the "DB2 - DB2" service

The J2EE role for accessing DB2 as specified in the Administrative Console

Symptom: Deadlocks when using compensation for a process

When you run a business process with compensation, a deadlock occurs on the table CONTEXTUALPROCLET.

This is caused by a missing index on the column COORDINATORHOME. Create an index on the table CONTEXTUALPROCLET. For example in DB2:

```
CREATE INDEX CP_COORDHOME ON CONTEXTUALPROCLET(COORDINATORHOME);
```

Symptom: Cannot deploy process template - template already exists

After reinstallation of the BPE container, process templates cannot be deployed. The error messages say that the process template already exists.

The process templates are saved in the BPEDB database. After reinstallation of the BPE container, the content of the database is not changed. All process templates that were deployed previously still exist in the database.

If you want to deploy a process template that was already deployed previously, you have to first delete this process template from the BPEDB in the table PROCESS_TEMPLATE_T.

Symptom: Activity description is not resolved

The description of a process activity that contains a substitution expression of the form `%variable.part[.xpathExpression]%` is not resolved at runtime.

The activity description, for example, in the audit trail, contains the original substitution expression and not the resolved activity description.

To solve the problem, check the WebSphere SystemOut.log file for exceptions and messages.

Common causes of the problem are:

- ▶ The specified variable or part does not exist.
- ▶ The expression contains periods “.” or percentage signs “%”. These characters are not allowed in substitution expressions.

Symptom: ClassNotFoundException, when trying to install a process application in an ND environment

For the installation of process applications in a cell, you have to put the database driver of the respective database system onto the classpath of the DeploymentManager process.

Note: This is not necessary if you have a single-server environment.

If you use a DB2 database as storage for the WebSphere Process Choreographer administration data, you have to add the full path of db2java.zip to the classpath. If you use a variable to specify the database driver location, make sure that this variable is actually defined on the DeploymentManager node.

You can change the classpath of the DeploymentManager process on the admin console:

1. On the navigator pane on the left-hand side of the admin console, expand SystemAdministration and click **DeploymentManager**.
2. A panel called dmgr is shown on the right-hand side. Under Additional Properties, click the **Process Definition** link.
3. On the next panel, under Additional Properties, click the **Java Virtual Machine** link.
4. Add the database driver location to the classpath input field and save your changes.

5. Shut down the DeploymentManager process and restart it before installing process applications.

Symptom: A timeout is received after invoking a JMS-based service from a non-interruptible process (microflow)

This problem occurs if the you have not installed the latest WSIF fixes.

If you want to invoke JMS-based services from non-interruptible processes, you must install the following WSIF fix:

- ▶ WAS-WebServices-WSIF_04-17-2003_5.0.1-5.0.0._cumulative_Fix.readme
- ▶ WAS-WebServices-WSIF_04-17-2003_5.0.1-5.0.0._cumulative_Fix.jar

Important: This fix is included in WebSphere Application Server 5.0.1 and later versions.

Now, when JMS-based services are invoked from non-interruptible processes, the current transaction for the non-interruptible process is suspended, the JMS request message is put to the message queue, the JMS reply message is received, and the transaction for the non-interruptible process is resumed.

Symptom: Cannot open the Process Choreographer Web client in the browser

This error occurs if you have the Administrative Console installed and the application server's classloader policy is set to `Single` (click **Servers** -> **Application Servers** -> **server1** -> **Application classloader policy**). `Single` is not supported in this setup, because the Administrative Console is based on Struts V1.0 and the Process Choreographer Web client is based on Struts V1.1. These Struts versions are incompatible and must not be on the same classpath. Use classloader policy `Multiple` to avoid the problem.

Verify, in the SystemOut.log file, that the following exception is logged.

Example 8-2 SystemOut.log with Exception

```
[7/17/03 14:38:18:750 CEST] 208bd83 WebGroup      I SRVE0180I:
[processportal.war] [/bpe] [Servlet.LOG]: action: Initializing configuration
from resource path /WEB-INF/struts-config.xml
[7/17/03 14:38:18:797 CEST] 208bd83 SystemOut      0 register('-//Apache
Software Foundation//DTD Struts Configuration 1.0//EN', 'wsjar:file:/C:/Program
Files/WebSphere/AppServer/installedApps/FMTCFUSSI/adminconsole.ear/struts.jar!/
org/apache/struts/resources/struts-config_1_0.dtd')
```

```

[7/17/03 14:38:18:797 CEST] 208bd83 SystemOut      0 register('-//Sun
Microsystems, Inc.//DTD Web Application 2.2//EN', 'wsjar:file:/C:/Program
Files/WebSphere/AppServer/installedApps/FMTCFUSSI/adminconsole.ear/struts.jar!/
org/apache/struts/resources/web-app_2_2.dtd'
[7/17/03 14:38:18:797 CEST] 208bd83 SystemOut      0 register('-//Sun
Microsystems, Inc.//DTD Web Application 2.3//EN', 'wsjar:file:/C:/Program
Files/WebSphere/AppServer/installedApps/FMTCFUSSI/adminconsole.ear/struts.jar!/
org/apache/struts/resources/web-app_2_3.dtd'
[7/17/03 14:38:18:812 CEST] 208bd83 SystemOut      0 resolveEntity('-//Apache
Software Foundation//DTD Struts Configuration 1.1//EN',
'http://jakarta.apache.org/struts/dtds/struts-config_1_1.dtd')
[7/17/03 14:38:18:812 CEST] 208bd83 SystemOut      0 Not registered, use
system identifier
[7/17/03 14:39:01:469 CEST] 208bd83 ServletInstan E SRVE0100E: Did not realize
init() exception thrown by servlet action: javax.servlet.UnavailableException:
Input/output error reading configuration from resource path
/WEB-INF/struts-config.xml
    at org.apache.struts.action.ActionServlet.init(ActionServlet.java:468)
    at javax.servlet.GenericServlet.init(GenericServlet.java:258)
    at com.ibm.ws.webcontainer.servlet.StrictServletInstance.doInit
(StrictServletInstance.java:82) at
com.ibm.ws.webcontainer.servlet.StrictLifecycleServlet._init
(StrictLifecycleServlet.java:147) at
com.ibm.ws.webcontainer.servlet.PreInitializedServletState.init
(StrictLifecycleServlet.java:270)

```

Symptom: Encountered error BPEE0031E while using the WebSphere Process Choreographer Web client

Error BPEE0031E can occur when a user tries to start a process of the process template MYTEMPLATE in the Web client, but the Enterprise Application MYAPPLICATION that contains the process template MYTEMPLATE is not started. You may receive an error similar to the following:

```
BPEE0031E: An error occurred in a data plug-in
```

Verify that the Enterprise Application is running.

1. Open the Administrative Console.
2. Identify the Enterprise Application that contains the process template MYTEMPLATE.
3. Make sure that this Enterprise Application is running.

Symptom: Encountered error BPEU0024E while using the WebSphere Process Choreographer Web client

If you receive an error like the following, the issue could be any one of three problems.

```
BPEU0024E: Could not establish a connection to local business process EJB.  
Reason: ''Naming Exception''
```

1. Your business process container might not be installed and configured. If this is the problem, resolve in the following way
 - a. Open the WebSphere Administrative Console and select **Servers -> Application Servers**. A list of all server instances appears.
 - b. Click the server on which you want to install and configure a business process container.
 - c. In the Additional Properties section, select **Business Process Container**.
 - d. At the bottom of the screen, select **Business Process Container Install Wizard**.
 - e. Follow the instructions on the screen to install and configure a business process container.
2. Your business process container might not be up and running properly. If this is the problem, resolve it in the following way:
 - a. Open the WebSphere Administrative Console and select **Applications -> Enterprise Applications**. A list of all installed applications appears.
 - b. Check if the application BPE_Container_<host>_<server> is started (where <host> is the name of the host where your business process container is installed and <server> is the name of the WebSphere server where your container is installed).

If the BPE_Container_<host>_<server> application is not started:
 - i. Click the check box next to the application.
 - ii. Click **Start** at the top of the list of enterprise applications.
3. Your naming service might not be up and running properly.
 - a. Open the WebSphere Administrative Console and select **Servers -> Application Servers -> <server> -> Server Components -> Name Server** (where <server> is the name of your WebSphere server instance).
 - b. The panel with the name server properties appears. On that panel, check the properties of your name server.

Symptom: Encountered error BPEU0002E while using the WebSphere Process Choreographer Web client

Check the following:

1. Make sure that the name saved in the worklist has the BPESystemAdministrator role defined.
2. If the name used for your worklist already exists for another worklist, use another name when you save the worklist.

Symptom: BPEContainer startup failed

Verify SystemOut.log during startup of the WebSphere Application Server.

Example 8-3 SystemOut.log with exception

```
W NMSV0605W: A Reference object looked up from the context "java:" with the
name "comp/env/jms/BPECF" was sent to the JNDI Naming Manager and an exception
resulted. Reference data follows:
```

```
Reference Factory Class Name: com.ibm.ws.util.ResRefJndiLookupObjectFactory
```

```
Reference Factory Class Location URLs: <null>
```

```
Reference Class Name: java.lang.Object
```

```
Type: ResRefJndiLookupInfo
```

```
Content: com.ibm.ws.util.ResRefJndiLookupInfo@b0956ee ResRefJndiLookupInfo:
```

```
Look up Name="jms/BPECF";JndiLookupInfo: jndiName="jms/BPECF"; providerURL="";
```

```
initialContextFactory=""
```

```
Exception data follows:
```

```
javax.naming.NameNotFoundException: jms/BPECF
```

```
at
```

```
com.ibm.ws.naming.jndicos.CNContextImpl.doLookup(CNContextImpl.java:1504)
```

To solve this problem, reinstall IBM WebSphere MQ and select **Java Messaging** during the custom install.

Symptom: javax.naming.NameNotFoundException: jms/BPECF

Either the external WebSphere MQ was installed with the default installation, which does not install MQ Java and JMS support, or the .jar files for WebSphere MQ's JMS implementation are missing from the classpath. These .jar files can be found in the java/lib subdirectory of the WebSphere MQ root directory.

Verify that SystemOut.log contains a javax.naming.NameNotFoundException: jms/BPECF, even though WebSphere's Administrative Console shows that the

connection factory BPECF is configured correctly under the WebSphere MQ JMS Provider.

The exception stack for this exception contains a `com.ibm.websphere.naming.CannotInstantiateObjectException` exception.

To solve this problem, install WebSphere MQ using a custom installation and make sure that WebSphere MQ Java and JMS support are selected.

Make sure that WebSphere can find the .jar files for WebSphere MQ's JMS implementation. Usually, the .jar files are found if the WebSphere variable `MQJMS_LIB_ROOT` points to the `java/lib` subdirectory of the WebSphere MQ installation directory, and the variable `MQ_INSTALL_ROOT` points to the root directory for WebSphere MQ. You can find these variables in the WebSphere Administrative Console by clicking **Environment -> Manage WebSphere Variables**, usually in the node scope.

Symptom: Cannot see my work items

This problem can occur for various reasons. These can include the following:

- ▶ The case-sensitivity of the logon user name is different from that defined for the user in the user registry.

WebSphere supports user registries for authenticating users. The type of user registry determines whether authentication via the user name is case-sensitive. When the local operating system user registry is activated, you have to specify the operating system user name and password.

Note: On Windows platforms, the specified user name is not case-sensitive, for example, user `User1` can successfully log on as `user1`.

Process Choreographer also uses the user registry for authorization. Thus, if there are work items stored for `User1` and you are logged on as `user1` using the Web Client or APIs, you will not be authorized to retrieve the work items and they will not be displayed. Therefore, it is recommended that you always log on to the WebSphere Application Server with the same user name. This is also true for other case-insensitive user registries.

- ▶ Staff resolution has failed.

The work items have not been created because staff resolution has failed. This can happen, for example, if the LDAP server cannot be reached due to network problems. Check the `System.out.log` file in your WebSphere Application Server installation directory for further information.

- ▶ On Windows, there are problems with the user registry.

If the Group Members staff verb is used with the user registry on Windows platforms, the group name must be fully-qualified, for example, MYCOMPUTER\mygroup, otherwise the users belonging to that group cannot see work items for that group.

Important: The above problem is resolved with Fix Pack 1 (WebSphere Application Server Enterprise Version 5.0.1). If you are experiencing this problem, upgrade to Fix Pack 1.

This hint replaces the 5.0.0 release note entry Logging in without a fully-qualified group name might not enable users to see work items.

Symptom: "Resource reference could not be located" message in SystemOut.log

The message is generated whenever access to the Process Choreographer takes place from a container environment where the corresponding resource references are not set in the deployment descriptor.

In particular, this can occur for operations triggered in the Administrative Console, for example, installing and uninstalling business process applications, or starting and stopping process templates.

A message like the following is printed to the SystemOut.log when using Process Choreographer.

Example 8-4 Error message

```
Resource reference cell/nodes/hostxy/servers/server1/jdbc/BPEDB could not be
located, so default values of the following are used: [Resource-ref settings]
res-auth:                1 (APPLICATION)
res-isolation-level:     0 (TRANSACTION_NONE)
res-sharing-scope:       true (SHAREABLE)
res-resolution-control:  999 (undefined)
```

This can give the impression that there is a configuration or installation problem.

Note: This is only an informational message which will not harm processing; you can simply ignore such messages.

Symptom: Nullpointer Exception is written into trace.log, but all processes complete successfully

Process Choreographer uses JMS messages to navigate through multi-transacted processes. If one of these messages cannot be processed, the transaction has to be rolled back. The Process Choreographer implements a Quiesce/Resume algorithm to make sure that messages are copied to a hold queue, if and only if they are poisoned.

In short, the algorithm works as follows:

1. The message is tried three times.
2. If the message fails to process three times, it is removed from the BPEIntQueue and copied to the BPERetQueue. The latter queue works as a buffer. If the buffer contains more than a configurable number of messages, the system switches to quiesce mode (the infrastructure seems to fail, because there were a number of messages in a row that could not be processed).
3. In quiesce mode and in normal processing mode, the next message on the queue is processed. If that works, the transaction can be committed, and all messages in the retention queue are copied back to the BPEIntQueue for a retry (if the system was in quiesce mode, the successful processing of the last message signals that the infrastructure is working again. The system switches back to processing mode).
4. Messages that have passed the retention queue for a configurable number of times are assumed to be poisoned and are copied into the BPEHoldQueue.

It should not happen that messages are left in the retention queue because they could not be processed earlier, and any instance data of the process the message belongs to is deleted; we would get the following scenario:

1. An interruptable process is started.
2. Messages are put in the BPEIntQueue.
3. If one of these messages is processed successfully, the message from the retention queue is copied into the BPEIntQueue and therefore processed later.
4. The NullpointerException is caused by the fact that the workflow engine needs a context for the message. If this context does not exist for any reason, the message cannot be processed. Note that this is really an exception, because the instance in which there is no context in the database but messages are left in the system should not take place!

Note: The exception is not caused by the current test case, which completed successfully. The message that causes the exception belongs to a process that is already deleted in the database.

Proposed solution

If there are only a few poisoned messages in the system, do nothing. The quiesce/resume algorithm will filter them out soon. This can be seen in the trace.log: the retry counter is incremented and the message is copied to the hold queue.

Example 8-5 Retry counter

```
Retry Count was set to 1 on EngineMessage  
Retry Count was set to 2 on EngineMessage.  
Retry Count was set to 3 on EngineMessage.
```

In test environments, it might be better to simply empty all queues and restart the test case that caused the exception. It will then work without any problems.

The queues can be emptied using the following WebSphere MQ commands:

```
dis qlocal('WQ_BPERetQueue') CURDEPTH  
clear qlocal('WQ_BPERetQueue')
```

Symptom: Concurrent Access to Variables in Process Choreographer

Variables are used in Process Choreographer to store messages in a business process.

Java snippets can be used to read and modify the contents of variables using the following API functions.

```
get<VariableName>  
set<VariableName>
```

If a business process is modeled in such a way that two parallel Java snippets access the same variable, database deadlocks can occur.

Variables are "local" to a business process instance. Thus, concurrent access to a variable can only take place within a single process instance, not between different instances of the same process model (to be more precise, this is true because the database system uses row-level locking where two different rows that are locked do not affect each other, generally speaking).

When reading a variable via `get<VariableName>`, a "read only" SQL query is run, producing a shared (S-) lock for the corresponding row in the database table.

If this lock has been obtained successfully, no other concurrent transaction can modify the value until the transaction is committed.

Two parallel Java snippet transactions accessing (reading) the same variable will not have a concurrency problem at all.

If a variable is to be modified in a Java snippet using `set<VariableName>`, a `SELECT ... FOR UPDATE` SQL query is run to acquire an update (U-) lock for the corresponding row in the database table. A following update statement acquires an exclusive (X-) lock for the row to prevent parallel reading of the variable and reduce the risk of a deadlock situation.

Two parallel Java snippet transactions with one reading a variable and another modifying might cause a lock wait situation until the reading completes.

However, if both Java snippets want to modify (update) the same variable, a database deadlock can occur: if the time between reading and updating the variable is long enough and at a certain point in time, both transactions read the variable and acquired S-locks, a deadlock will occur because both transactions are waiting for the other to get the U- and X- locks for modifying the variable contents. As a result, one transaction will be rolled back (and will be retried when running a long-running macroflow).

Because deadlocks significantly affect performance, try to avoid patterns in your business process where two parallel Java snippets modify the same variable: either add an empty activity in one branch to make sure both Java snippets do not run at the very same time or add additional variables, one for each Java snippet.

Symptom: People Activity gets in State STOPPED

When running a process with the WebSphere Process Choreographer, a people activity appears to be in state STOPPED. Verify the error message; it should show something like the following:

```
com.ibm.bpe.api.EngineUncaughtExceptionInActivityException: BPEE0001E:  
Uncaught exception in activity 'AnActivity'.
```

This could be due to an improper expiration time format. Verify that the expiration time/date defined for the people activity is in the right format. No blanks are allowed between the numbers and the unit of time.

Symptom: Syncpoint Manager Exceptions when accessing DB2 for z/OS

This can happen when WebSphere Application Server attempts to start the first XA transaction with a remote DB2 for z/OS database.

Open SystemOut.log file and check to see whether you have logs similar to the following.

Example 8-6 SystemOut.log with exception

```
[2/7/03 8:23:09:165 CET] 5a3ff7ab Engine          E com.ibm.bpe.engine.Engine
TRAS0014I: The following exception was logged COM.ibm.db2.jdbc.DB2Exception:
[IBM][CLI Driver][DB2] SQL30090N Operation invalid for application execution
environment. Reason code = "1". SQLSTATE=25000
    at COM.ibm.db2.jdbc.app.SQLExceptionGenerator.throwSQLException
(SQLExceptionGenerator.java:270)at
COM.ibm.db2.jdbc.app.SQLExceptionGenerator.throwSQLException
(SQLExceptionGenerator.java:207)at
COM.ibm.db2.jdbc.app.SQLExceptionGenerator.check_return_code
(SQLExceptionGenerator.java:458)at
COM.ibm.db2.jdbc.app.DB2PreparedStatement.execute2
(DB2PreparedStatement.java:2110)at
COM.ibm.db2.jdbc.app.DB2PreparedStatement.executeUpdate
(DB2PreparedStatement.java:1642)at
com.ibm.ws.rsadapter.jdbc.WSJdbcPreparedStatement.executeUpdate
(WSJdbcPreparedStatement.java:578)at
com.ibm.bpe.database.TomInstanceCache.flushDirtyObjects
(TomInstanceCache.java:131)at com.ibm.bpe.database.Tom.beforeCompletion
(Tom.java:667)at com.ibm.bpe.admin.AdminService.installProcessModule
(AdminService.java:307)at
com.ibm.bpe.processarchive.ProcessModuleInstallTask.performTask
(ProcessModuleInstallTask.java:110)at com.ibm.ws.management.application.
SchedulerImpl.run(SchedulerImpl.java:216)at
java.lang.Thread.run(Thread.java:512)
```

SQL code 30090N indicates problems with the syncpoint manager. These exceptions seem to be caused by invalid syncpoint manager log entries in the sqllib/spmlog directory. Perform the following steps:

1. Clear the entries in the sqllib/spmlog directory and restart the server.
2. You may also have to increase SPM_LOG_FILE_SZ; check the product documentation.
3. Make sure that the instance configuration variable SPM_NAME points to the local machine with a hostname not longer than eight characters. If the hostname is longer than eight characters, define a short alias in /etc/hosts.

Note: The syncpoint manager is a component of the local DB2 Connect™ system. It is responsible for managing distributed transactions with a remote DB2 host system.

Symptom: Installing an enterprise application results in the error "BPED0000E: The process model ... already exists" because a previous install of the same application was cancelled

When you try to install an enterprise application, you may get an error message similar to the following:

```
com.ibm.bpe.plugins.DeploymentDuplicateProcessModelException:  
BPED0000E: The process model with name 'ProcessModelName' and valid from  
date 'Tue 2002-01-01 12:00:00.000' already exists.
```

This will happen if the installation of an enterprise application is cancelled and all process templates contained in the application have already been saved in the Process Choreographer database and are not deleted.

When you install an enterprise application that contains business process modules (.far files), the process templates are always immediately written to the Process Choreographer database. If you decide to cancel the installation by not saving your configuration changes, the enterprise application is not installed but the templates remain in the database.

This problem occurs in both of the following circumstances:

- ▶ If you installed the application using the Administrative Console, and clicked **Cancel** in the Save to Master Configuration panel.
- ▶ If you used `wsadmin $AdminApp install` to install the application, and issued the command `wsadmin $AdminConfig reset`.

If you have cancelled an installation, you must remove the process templates from the table `PROCESS_TEMPLATE_T` in the Process Choreographer database.

In order to remove the entry from the database, follow the steps below:

1. Connect to your Process Choreographer database; the default name is `BPEDB`.
2. If you do not know the application name that was used to install the enterprise application, issue the query:

```
select distinct APPLICATION_NAME from PROCESS_TEMPLATE_T where  
NAME='<ProcessModelName>' and VALID_FROM='<validFrom>'
```

Note: The format of the <validFrom> column depends on your locale. If you are not sure of the correct format for your locale, you can find it by issuing another query, for example:

```
select VALID_FROM from PROCESS_TEMPLATE_T where NAME='<ProcessModelName>'
```

3. Delete all templates associated with the application from the database by issuing the following command:

```
delete from PROCESS_TEMPLATE_T where APPLICATION_NAME='<application_name>'
```

Avoding this problem

When installing an enterprise application containing business processes modules, never cancel the installation. To ensure a consistent configuration, always save your configuration changes. Later, if you uninstall the application, the process templates will be removed from the Process Choreographer database correctly.

Symptom: Do not set CCSID if Transport Type is "Bindings"

Binding the J2EE resources BPECF and BPECFC for Process Choreographer fails with the error message Invalid configuration passed to resource binding logic. The WebSphere server cannot be started successfully.

Verify the SystemOut.log, and look for something similar to the next example.

Example 8-7 SystemOut.log with error

```
[1/30/03 0:52:48:213 CET] 12041c7c ResourceMgrIm I WSVR0049I: Binding
BPERetQueue as jms/BPERetQueue
[1/30/03 0:52:48:364 CET] 12041c7c ResourceMgrIm E WSVR0017E: Error encountered
binding the J2EE resource, BPECF, as jms/BPECF from resources.xml
com.ibm.ws.runtime.component.binder.ResourceBindingException: invalid
configuration passed to resource binding logic. REASON: Failed to create
connection factory at
com.ibm.ejs.jms.JMSConnectionFactoryReferenceable.<init>(JMSConnectionFactoryRe
ferenceable.java:127)at
com.ibm.ejs.jms.JMSResourceRefBuilderImpl.createNonGenericConnectionFactoryRefe
renceable(JMSResourceRefBuilderImpl.java:505)at
com.ibm.ejs.jms.JMSResourceRefBuilderImpl.createNonGenericQueueConnectionFactory
yReferenceable(JMSResourceRefBuilderImpl.java:331)at
com.ibm.ejs.jms.JMSResourceRefBuilderImpl.createMQQueueConnectionFactoryReferen
ceable(JMSResourceRefBuilderImpl.java:308)at
com.ibm.ws.runtime.component.binder.MQQueueConnectionFactoryBinder.getBindingOb
ject(MQQueueConnectionFactoryBinder.java:119)at
com.ibm.ws.runtime.component.ResourceMgrImpl.bind(ResourceMgrImpl.java:255)at
com.ibm.ws.runtime.component.ResourceMgrImpl.installResourceProvider(ResourceMg
```

```
rImpl.java:606)at
com.ibm.ws.runtime.component.ResourceMgrImpl.installResource(ResourceMgrImpl.java:573)at
com.ibm.ws.runtime.component.ResourceMgrImpl.installResources(ResourceMgrImpl.java:533)at
com.ibm.ws.runtime.component.ResourceMgrImpl.loadResources(ResourceMgrImpl.java:413)at
com.ibm.ws.runtime.component.ResourceMgrImpl.start(ResourceMgrImpl.java:362)at
com.ibm.ws.runtime.component.ContainerImpl.startComponents(ContainerImpl.java:343)at
com.ibm.ws.runtime.component.ContainerImpl.start(ContainerImpl.java:234)
    at
com.ibm.ws.runtime.component.ApplicationServerImpl.start(ApplicationServerImpl.java:117)at
com.ibm.ws.runtime.component.ContainerImpl.startComponents(ContainerImpl.java:343)at
com.ibm.ws.runtime.component.ContainerImpl.start(ContainerImpl.java:234)
    at com.ibm.ws.runtime.component.ServerImpl.start(ServerImpl.java:182)
    at com.ibm.ws.runtime.WsServer.start(WsServer.java:135)
    at com.ibm.ws.runtime.WsServer.main(WsServer.java:232)
    at java.lang.reflect.Method.invoke(Native Method)
    at com.ibm.ws.bootstrap.WSLauncher.main(WSLauncher.java:94)
```

To solve this problem, follow the steps below:

1. Log on with Administrative Console and go to **Resources -> WebSphere MQ JMS Provider -> WebSphere MQ Queue Connection Factories -> BPECF / BPECF**.
2. Check that the Transport Type is Bindings.
3. If so, clear the CCSID field. This is only necessary if the Transport Type is Client.
4. Save the changes and restart the application server.

8.1.2 Business rule beans

Business rule beans (BRBeans) are beans that allow you to separate business rules from your application. The eMerge scenario could create rules that categorize a customer's credit level and the credit policy could be changed based on a schedule or through user intervention.

Assuming that the code for accessing the business rule beans was properly added to your application, there are still some configuration aspects where things could go wrong. BRBeans use a database to store their rule information. You must set up a rule structure and refer to it correctly in the code that calls BRBeans.

Things to check

1. Make sure that a BRBeans.jar is installed on your application server.
2. You must have a business rule database created with a datasource for the BRBeans to use. If there are datasource problems, see the WebSphere Application Server section on data sources and JDBC Providers for troubleshooting information.
3. There are security roles assigned to the BRBeans EJBs. If security is enabled, assign users and groups to the BRBean roles. These roles are RuleUser, RuleManager, and DenyAllRole. If these roles are not assigned, you will see security errors. Your application must be secured correctly for credentials to be passed on to the EJB container. See the WebSphere Application Server section on application security for more information.
4. Review the JVM logs for business rule bean exceptions and use the Log Analyzer. To enable a BRBean trace, add `com.ibm.ws.brb.*=all=enabled:com.ibm.websphere.brb.*=all=enabled` in the trace specification in the Administrative Console.
5. BRBeans throw a `com.ibm.websphere.brb.BusinessRuleBeans` exception for exceptions that occur. Having your BRBean code do a `printStackTrace` when this exception is caught can help diagnose a problem based on the stack. Other exceptions that might be thrown are:
 - `ConstraintViolationException`.
 - `NoRuleFoundException`. Verify that the rules your application is trying to access actually exist in the BRBeans database. Use the `rulemgmt` GUI to view and manipulate rules. Check that your rules are not expired or pending if you are trying to use them.
 - `MultipleRulesFoundException`.
6. If you are having problems bringing up the `rulemgmt` GUI, make sure that your properties file contains the correct JNDI names and port. The port should be the bootstrap port of the server where BRBeans are installed. The default `brbeans` property file has the default port for an unfederated server1. If security is enabled, you will have to log in with an administrator ID.

Symptom: Database deadlock

In version 5.0, business rule beans could cause a deadlock if multiple clients access the same rules at the same time, if the rules were set up for remote firing. This has been fixed in V5.0.1.

Symptom: Rules not triggered

Look in the JVM logs for clues. You might see a null pointer exption if no rule was returned and the exception was not caught.

If you use `printStackTrace` on your `BusinessRuleBeansException`, then the `System.err` JVM log could contain the following.

Example 8-8 `printStackTrace` for `BusinessRuleBeansException`

Stack trace:

```
com.ibm.websphere.brb.NoRulesFoundException
  at com.ibm.websphere.brb.strategy.AcceptOneFilteringStrategy.filterRules
(AcceptOneFilteringStrategy.java:44)
  at com.ibm.websphere.brb.TriggerPoint.triggerGeneric(TriggerPoint.java:680)
  at com.ibm.websphere.brb.TriggerPoint.trigger(TriggerPoint.java:570)...
```

Use the `rulemgmt` GUI to review your rules and their use dates. Either a rule does not exist, it is still scheduled or it has expired. Add or change the rule. For more information, enable trace for business rule beans.

If you have SQL errors, review your `datasource` and `database` for problems with the business rule beans database.

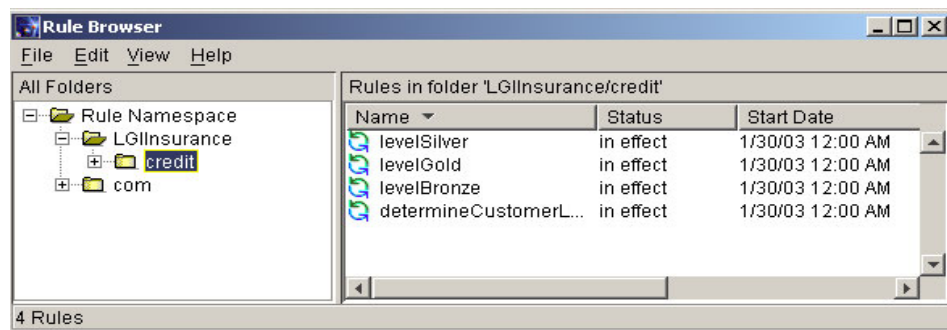


Figure 8-1 Business rule beans GUI

8.1.3 Extended messaging

Extended messaging, also sometimes referred to as *container managed messaging* (CMM), extends the base JMS support, support for EJB 2.0 message-driven beans, and the EJB component model to use the existing container-managed persistence and transactional behavior. You can separate your business logic from messaging code.

Assuming that your code was implemented properly, errors or failure can happen. Some of the failures may be similar in behavior to the regular JMS messaging failures. Extended messaging obviously deals with queues or topics. It can also deal with methods that you call with the receiver bean. These could be additional messages to send or calls to EJBs.

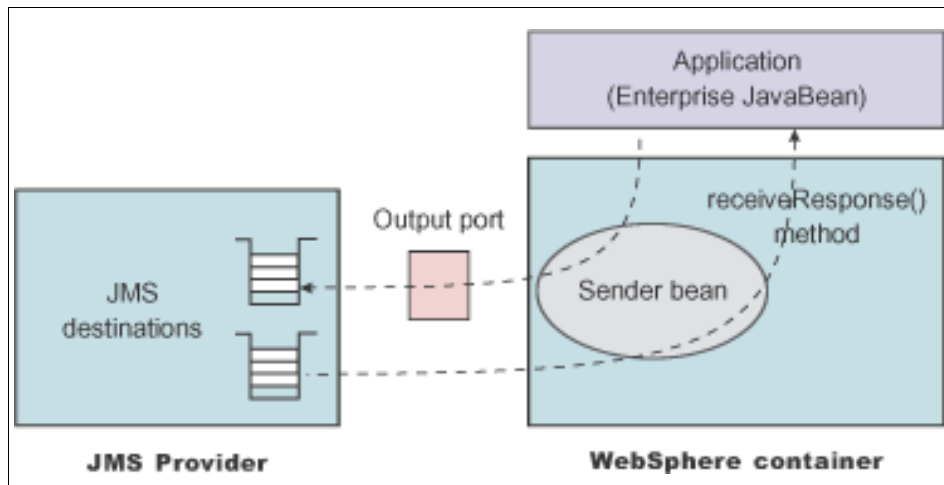


Figure 8-2 Figure from WebSphere InfoCenter with receiver message flow

Things to check

1. In the JVM logs, any extended messaging related errors start with extended messaging.
2. There are many resources that need to be configured on the WebSphere Application Server side. You need to create resources for an output port and either an input port or listener port. If you experience problems, you should confirm that all your settings are correct, including JNDI names, queues, queue managers, and any user IDs. Make sure that you are creating the right resources for your embedded messaging, whether it is internal or external MQ with a queue or topic.
3. If you use listener ports, they must be configured on all the application servers that you want to listen for messages. Check that the listener port is started.
4. The application may still start even if the MQ resources failed to connect to the queue or topic. If the application does not seem to be sending or receiving messages, check the JVM logs to confirm that they started normally.
5. If there is a problem with the queue, investigate your JMS provider or check your resources again to make sure that you are using the correct parameters.

6. If you use embedded messaging, make sure that the user ID starting the JMS server is fewer than twelve characters. On Unix, check that the user is part of the mqm and mqbrkr group. See WebSphere Application Server installation instructions for the correct embedded user setup.
7. If you have problems after receiving a message, check that any resources that the receiver bean needs are available. If it sends another message or calls another EJB, then the resources need to be set up for the code being called. For instance, if you have an SQL error, this is not necessarily an extended messaging error, but a problem with the datasource for the EJB that is being called to handle a message. At this point, you can confirm that you are receiving messages even if the message is not well handled.
8. If the sender and receiver were not created at the same time, check that there is data mapping for the message and that the sender and receiver are mapping the message in the same way. WebSphere Studio Application Developer Integration Edition has wizards to guide you through the creation process to make sure the sender and receiver beans match.
9. If you need to send a JMS text message instead of a stream message, you may have to write a custom formatter class to replace the CMMFormatter for your message. For example, sending an IMS™ message using the MQ-IMS bridge requires a JMS text messages. See Vernon Green's article on the Extended Messaging Formatter and Parser for more information at:
http://www-106.ibm.com/developerworks/websphere/techjournal/0401_green/green.html
10. To enable trace, select the **Messaging** group for the Trace Specification in the WebSphere Administrative Console. Use the messaging section in the WebSphere Application Server topic for more messaging-related troubleshooting.

Symptom: Application does not receive message

Check the JVM logs for a transaction rollback. If the error message says that the mapping does not match, you will have to correct the extended messaging bean mapping.

Check the JVM logs for MQ-related messages. For instance, this WebSphere Application Server message leads to an MQ error message.

Example 8-9 JVM SystemOut.log with linked MQ exceptions

```
[11/20/03 15:23:09:981 EST] 15cc809e FreePool      E J2CA0046E: Method
createManagedConnctionWithMCWrapper caught an exception during creation of the
ManagedConnection for resource JMS$GriffinConnect0I, throwing
ResourceAllocationException. Original exception:
```

```

javax.resource.spi.ResourceAdapterInternalException: createQueueConnection
failed
    at com.ibm.ejs.jms.JMSCUtils.mapToResourceException(JMSCUtils.java:123)
    at com.ibm.ejs.jms.JMSManagedQueueConnection.createConnection
(JMSManagedQueueConnection.java:119)
    at com.ibm.ejs.jms.JMSManagedConnection.<init>
(JMSManagedConnection.java:157)
    at com.ibm.ejs.jms.JMSManagedQueueConnection.<init>
(JMSManagedQueueConnection.java:64)
    ...
Next Linked Exception:
javax.jms.JMSEException: MQJMS2005: failed to create MQQueueManager for
'ogoshi.rchland.ibm.com:QM_ogoshi'
    at com.ibm.mq.jms.services.ConfigEnvironment.newException
(ConfigEnvironment.java:556)
    at com.ibm.mq.jms.MQConnection.createQM(MQConnection.java:1736)
    at com.ibm.mq.jms.MQConnection.createQMXA(MQConnection.java:1077)
    at com.ibm.mq.jms.MQQueueConnection.<init>(MQQueueConnection.java:123)
    at com.ibm.mq.jms.MQQueueConnection.<init>(MQQueueConnection.java:80)
    at com.ibm.mq.jms.MQXAQueueConnection.<init>(MQXAQueueConnection.java:46)
    at com.ibm.ejs.j2c.poolmanager.PoolManager.reserve(PoolManager.java:1590)
    ...
Next Linked Exception:
com.ibm.mq.MQException: MQJE001: An MQException occurred: Completion Code 2,
Reason 2059
MQJE011: Socket connection attempt refused
    at com.ibm.mq.MQManagedConnectionJ11.<init>
(MQManagedConnectionJ11.java:239)
    at com.ibm.mq.MQClientManagedConnectionFactoryJ11._createManagedConnection
(MQClientManagedConnectionFactoryJ11.java:276)
    at com.ibm.mq.MQClientManagedConnectionFactoryJ11.createManagedConnection
(MQClientManagedConnectionFactoryJ11.java:296)
    at com.ibm.mq.StoredManagedConnection.<init>
(StoredManagedConnection.java:80)
    ...

```

At this point, WebSphere MQ should be investigated. In the case of this error, the WebSphere MQ machine was powered down and this caused the MQJE011: Socket connection attempt refused error.

Review either the embedded messaging or WebSphere MQ sections.

If your application receives messages on a Listener Port, verify that the Listener Port is running.

Symptom: Message does not send

Review the JVM logs for error messages. There may be a problem with the queue being used.

Example 8-10 JVM SystemOut.log with a failure to enlist

```
[11/20/03 15:43:18:399 EST] 6e51c0a9 PrivExAction W J2CA0114W: No
container-managed authentication alias found for connection factory or
datasource JMS$GriffinConnectOI.
[11/20/03 15:43:18:839 EST] 6e51c0a9 XATransaction E J2CA0030E: Method enlist
caught javax.transaction.SystemException: Failed to start the transaction
association.
    at com.ibm.ejs.jts.jta.TransactionImpl.enlistResource
(TransactionImpl.java:782)
    at com.ibm.ejs.jts.jta.JTSXA.enlist(JTSXA.java:998)
    at com.ibm.ejs.j2c.XATransactionWrapper.enlist
(XATransactionWrapper.java:740)
    at com.ibm.ejs.j2c.ConnectionEventListener.interactionPending
(ConnectionEventListener.java:743)
    at com.ibm.ejs.jms.JMSManagedSession.interactionPending
(JMSManagedSession.java:968)
    ...
while trying to enlist resources from datasource
JMS$GriffinConnectOI$JMSManagedConnection@1887518856 with the Transaction
Manager for the current transaction, and threw a ResourceException.
[11/20/03 15:43:18:929 EST] 6e51c0a9 ConnectionEve A J2CA0056I: The Connection
Manager received a fatal connection error from the Resource Adaptor for
resource JMS$GriffinConnectOI. The exception which was received is
javax.jms.JMSEException: Enlist failed
```

In this case, you could enable trace to look for further information. You should check on the status of your WebSphere MQ. In this case, the WebSphere MQ machine was down.

If you are using an WebSphere MQ Broker, you might see a message like the following when the broker cannot be contacted.

Example 8-11 WebSphere MQ Broker machine not responding

```
[11/21/03 13:24:40:475 EST] 553d9a26 ConnectionEve A J2CA0056I: The Connection
Manager received a fatal connection error from the Resource Adaptor for
resource JMS$GriffinConnectPub. The exception which was received is
com.ibm.mq.jms.NoBrokerResponseException: MQJMS5053: *** No broker response.
Please ensure that the broker is running. If you are using the WebSphere MQ
broker check that your brokerVersion is set to V1 ***
[11/21/03 13:24:40:806 EST] 553d9a26 MDBListenerIm W WMSG0019E: Unable to start
MDB Listener GriffinSubscribe, JMSDestination jms/GriffinDestPub :
com.ibm.mq.jms.NoBrokerResponseException: MQJMS5053: *** No broker response.
```

```
Please ensure that the broker is running. If you are using the WebSphere MQ
broker check that your brokerVersion is set to V1 ***
    at com.ibm.mq.jms.MQBrokerSubscriptionEngine.getBrokerResponse
(MQBrokerSubscriptionEngine.java:3303)
    at com.ibm.mq.jms.MQBrokerSubscriptionEngine.openSubscription
(MQBrokerSubscriptionEngine.java:317)
    ...
---- Begin backtrace for Nested Throwables
com.ibm.mq.MQException: MQJE001: Completion Code 2, Reason 2033
    at com.ibm.mq.MQQueue.getMsg2(MQQueue.java:893)
    at
com.ibm.mq.jms.MQBrokerSubscriptionEngine.getBrokerResponse(MQBrokerSubscriptio
nEngine.java:3296)
    at
com.ibm.mq.jms.MQBrokerSubscriptionEngine.openSubscription(MQBrokerSubscription
Engine.java:317)
```

8.1.4 Asynchronous beans

Asynchronous beans allow applications to do work in parallel and create listeners or alerts. In the case of the eMerge scenario, we may want to split off some work to compare findings in the two insurance databases.

Asynchronous beans can involve many components. They might have a work manager and a listener and might work with a variety of other resources. It is important to discern between the misbehavior of an asynchronous bean and misbehavior of one of the resources it works with, such as a data source. There could be problems with how the asynchronous bean works that prevent the datasource from being used correctly.

Things to check

1. Check the JVM logs for errors related to the asynchronous beans.
2. Asynchronous beans that perform work need a work manager. Check that your work manager is properly configured.
3. If your application shares asynchronous scope objects, then all of the sharing components need the same work manager.
4. Check that your context is correct. If your bean inherits from the Internationalization context, it needs to be a sticky context on your work manager.
5. Be careful to check that your asynchronous beans are looking up resources and using them correctly. You can cache a connection factory, but not a connection. See the WebSphere InfoCenter for information on asynchronous

beans and for examples of the correct and incorrect use of bean connection management (*Example: Asynchronous bean connection management*).

6. To enable trace, select the **AsynchBeans** and **AysnchBeans_Alarms** groups for the Trace Specification in the Administrative Console.

8.1.5 Dynamic query

Dynamic query is a useful feature for performing queries that the EJB query language for EJB finders will not let you easily perform on finders. If you use dynamic query inappropriately, there will be a performance hit. If you use dynamic query to replace complicated use of EJB query finders and traverse container manager relationships or to use SQL statements not available in the EJB query language, then it can be very useful.

Dynamic query uses an application called query.ear to do work and interact with your applications EJBs.

The eMerge scenario could use dynamic query if desired to add a query on EJBs that includes an aggregation function such as SUM or AVG.

Things to check

1. The EJBs that will be used in dynamic queries must be at the 2.0 EJB specification. If you have existing 1.1 EJBs that you want to use, they must be migrated to 2.0 or recreated as new 2.0 beans. If you use WebSphere Studio Application Developer, you can use the migration tools available and find the steps for migration in the help files under *Migrating enterprise bean code from Version 1.1 to Version 2.0* from the product help.
2. If you are not using an application profile for the EJBs that will be accessed with dynamic query, then you may need to change the access intent policy, depending on the database that you use.
3. Verify that the data source and data source mappings assigned to the EJBs that are being queried are correct. Review the WebSphere Application Server section on data sources and applications for more troubleshooting suggestions.
4. The application query.ear needs to be installed on the server on which your application with dynamic query is running.
5. On a 5.0 install of WebSphere Application Server Enterprise, your Application Class Loader policy needs to be set to SINGLE instead of MULTI in order to use the query bean interfaces. This has been fixed at later Fix Pak levels.

6. Make sure that your query is accurate. You need to call beans and methods that exist. Your query must be well formed. See the WebSphere InfoCenter on dynamic query for examples, such as the article *SELECT clause*.

Beware of using reserved words in your query. These words are reserved for WebSphere Application Server Enterprise dynamic query: all, as, distinct, empty, false, from, group, having, in, is, like, select, true, union, where and identifiers that start with an underscore (for example: `_blue`).

7. Review the JVM logs for clues and the Log Analyzer for more information. To trace dynamic query actions, use `com.ibm.websphere.eexquery.*=all=enabled` in the Trace Specification field in the Administrative Console.

Symptom: Query fails

Check the JVM logs. You may find the following message.

Example 8-12 Dynamic query exception in JVM log

```
com.ibm.websphere.csi.CSException: unable to get EJBObject; nested exception
is: com.ibm.websphere.csi.CSException: unable to get wrapper; nested exception
is: java.lang.ClassCastException: java.lang.Byte
```

In this case, you have a problem where the CMP EJB field of type `java.lang.Boolean` is mapped to an IBM Cloudscape or MSSQLServer column of type `tinyint`. Dynamic query does not handle this mapping if you used meet-in-the-middle mapping or the default mapping created by the `ejbdeploy` tool. To work around this problem, change your CMP attribute from `java.lang.Boolean` to the java primitive type `boolean`. A second option is to change the column type from `tinyint` to `smallint`. For more information, see the Technote *Dynamic Query runtime ClassCastException due to bad mapping of java.lang.Boolean to tinyint* on the WebSphere Application Server Enterprise support site.

Review the JVM log for dynamic query messages that start with `WQRY`, and their associated message. An incorrect field name would produce an error like this one.

Example 8-13 Dynamic query error

```
WQRY0036E: Book o does not have a field Bi_ID
```

8.1.6 Scheduler

The scheduler is a WebSphere Application Server Enterprise feature that allows you to schedule tasks to occur at certain times and repeat as desired. The eMerge scenario does not use this feature, but if it had tasks that it wanted to perform without user intervention, such as running statistics or accessing a Web service routinely, the scheduler could do this.

The scheduler itself uses a database to store tasks and a scheduler service. It can interact with a variety of other components, depending on what kind of task it performs.

Things to check

1. If you have problems accessing your scheduler database, see the data source and JDBC provider sections for more troubleshooting tips. Look in the WebSphere Infocenter for more information on how to set up your scheduler database correctly. The article *Using the scheduler service* is a good place to start.
2. Confirm that your scheduler configurations settings are correct. Check that you are using the correct data source JNDI name and alias. Check that your polling interval has the correct value if you do not see your tasks occurring on time.
3. Verify that the scheduler service is enabled on your server. You can check this by clicking **Servers** -> **<server_name>** -> **Scheduler Service**.
4. You may have to enable last participant support for your application using the scheduler if it is triggering a one-phase commit resource.
5. If you have a mixed version cluster, there may be interoperability problems with the scheduler. For versions at release 5.0, you should apply fix packs associated with interim fixes PQ72184 and PQ72742 to interoperate with version 5.0.1 and later. See the WebSphere InfoCenter article *Interoperating with the Scheduler service* for more information.
6. Look for errors in the JVM logs and use the Log Analyzer. To enable a trace of the scheduler, enter Scheduler=all=enabled for the Trace Specification in the Administrative Console.

Example 8-14 Scheduler messages in the JVM log on a successful start

```
SCHD0036I: The Scheduler Service is initializing.  
SCHD0037I: The Scheduler Service has been initialized.  
SCHD0031I: The Scheduler Service is starting.  
SCHD0032I: The Scheduler Instance sched/samples/scheduler/accountreport is  
starting.
```

The Scheduler Daemon for instance sched/samples/scheduler/accountreport has started.

The Scheduler Instance sched/samples/scheduler/accountreport has started.

SCHD00011: The Scheduler Service has started.

These messages indicate that the scheduler service started normally and initialized the scheduler instances.

7. Review PMRs on the WebSphere Application Server Enterprise support site. There are scheduler fixes in version 5.0.1 and 5.0.2.

Note: Keep in mind that the scheduler may be working normally, but the task that it is running could fail. For instance, if the scheduler sends a message to a queue, the resource provider for the queue could be incorrectly configured or there could be something wrong with the queue itself.

Symptom: Scheduler does not run scheduled task

Review the JVM log at server startup to confirm that the scheduler service started normally.

Review the schedule for the task and make sure that it has the correct timing and number of repetitions with an accurate polling time.

Symptom: Schedule task fails

Check the JVM log for this error message on a transaction rollback.

Example 8-15 One-phase commit resource tries to participate with two-phase

WTRN0063E: An illegal attempt to commit a one phase capable resource with existing two phase capable resources has occurred.

The transaction with the scheduler and the task it calls is a two-phase commit transaction. There is a one-phase capable resource in the transaction. You need to enable last participant support for your application to allow the one-phase capable resource.

Look for messages related to the task that the scheduler called. There may be errors associated with the resources for the task instead of with the scheduler itself.

Symptom: Scheduled task does not run on time

Check your polling interval for the scheduler. It may not be short enough for your task.

The scheduler runs the task on time relative to the poll interval, the number of alarm threads available and the server load. If the alarm thread limit is reached, then the tasks are queued until threads become available.

Adjust the poll interval, the number of alarm threads on the work manager and balance the server load to increase the scheduler's accuracy.

8.1.7 Startup beans

Startup beans start with your application and execute work when your application starts up and stops.

There are some problems that could occur with startup beans. There could also be problems with whatever task the startup beans are performing. This could include EJB calls, using JMS services, Web services, or whatever work your startup bean has been assigned.

Things to check

1. Review the log for any problems the startup beans may have had during application startup or stop.
2. For the startup beans start() and stop() methods, you cannot use the TX_MANDATORY property, but you can use any other TX_* type. If you use TX_MANDATORY, you will see an error and the application will not start.
3. If you use the optional environment property, StartupPriority, make sure it is the right type of integer and set to a valid value. If it is incorrect, the application will not start.

Symptom: Application does not start

If there is something wrong with your startup beans, then the application will not start. Review the logs for errors messages relating to the startup beans.

8.1.8 Application profiling

Application profiles allow you to define access intent policies for back-end access and selectively apply them to tasks in your application. A thorough understanding of the application is required to correctly create and change application profiles.

The eMerge scenario could use application profiles to differentiate between a customer viewing their policy where no change is being made and an employee updating a policy where the record needs to be locked.

Things to check

1. If you are not getting correct results with your application and are experiencing problems with your database or information in the database, review your application profile to determine whether tasks are using the wrong access intent policy.
2. If you add or edit task settings on the application, make sure that you do not assign more than one application profile to a single task within an application. Your application cannot be restarted with more than one application profile configured to a task.
3. To enable trace, use the trace specification: `RRA=all=enabled`.
4. Make sure that your method0-level access intents are set to the default intent. Do not mix method-level access intent configuration and application profiling.

Symptom: Unexpected concurrency/locking strategy

Review your application profile settings. Make sure that the task/transaction match, the entity is configured with the task and also configured with the right intent.

8.1.9 Object pools

Object pools allow Java objects to be pooled and reused by EJBs and servlets.

Things to check

1. Are you trying to pool JDBC or JMS connections? You should not pool these as WebSphere Application Server pools; these are connections with specialized code.
2. You can only look up objects in object pools with EJB or Web containers. You cannot look them up with J2EE application clients.
3. Review your object pool manager settings.
4. Is your object cleaned out properly before being returned to the pool?
5. To enable trace, use the trace specification:
`com.ibm.ws.asynchbeans.pool.*=all=enabled`

Symptom: Cannot reuse an object

Review your code and verify that the objects are being returned to the pool. If they are not returned, they cannot be reused.

Symptom: Synchronization or deadlock problems

You can implement an object thread pool as thread-safe or not. A non-thread-safe pool is faster, but not synchronized. If there are two threads using the pool at the same time, you should use a thread-safe pool.

8.1.10 WorkArea service

The WorkArea service allows users to pass along information between their application components without needing to create extra parameters to pass information along.

The eMerge scenario could use this feature to pass along user account information between methods if there is information that makes passing by parameter awkward.

Things to check

1. Does the client that creates the work area terminate it?
2. If you are on V5.0 or V5.0.1, review the WebSphere Application Server support site for work area related APARs.
3. Review APAR PQ76492 for a 5.0.2 related program defect: *In PME 502 Work Area service the values for MaxSendSize and MaxReceiveSize got hard coded to their default values*
4. To enable trace, use the trace specification
`com.ibm.ws.workarea.*=all=enabled.`

Symptom: Remote invocations associated with WorkArea context may unexpectedly fail

If you see `com.ibm.ws.javax.activity.SystemException` or an `org.omg.CORBA.IMP_LIMIT` error then you may need to apply APAR PQ76492, which deals with the maximum send and a maximum receive values being hardcoded.

8.1.11 Internationalization service

The internationalization service allows your application to use the internationalization context APIs to manage locale and time zone information.

Things to check

1. Review the JVM logs for error messages. Look for messages starting with I18N****.
2. Check that the internationalization service is enabled on your server. It can be enabled by going to **Server -> Application Servers -> servername -> Internationalization Service**. If you enable or disable the service, you have to restart the server for the changes to take effect.
3. For EJB and Web service enabled clients, the i18nctx.jar file needs to be in the classpath used by the launchClient tool. This should be done by default, but if someone disabled the service by removing the i18nctx.jar from the lib directory, it will need to be replaced.
4. To enable trace for the internationalization service, use these trace strings in the Trace Specification box on the Administrative Console:

```
com.ibm.ws.i18n.context.*=all=enabled:com.ibm.websphere.i18n.context.*=all=enabled
```

Symptom: Application fails, JNDI lookup error in the log

You will see a JNDI lookup exception from the internationalization service if the service is not enabled.

Symptom: Application fails

Review your application's deployment descriptor internationalization policies. If they are missing or corrupt, you will see problems with the application. For more information, review the internationalization trace using `com.ibm.ws.i18n.context.*=all=enabled`.

8.1.12 Last participant support

Last participant support allows you to have a one-phase commit resource participate in a two-phase transaction with other two-phase commit resources. You might use this with other WebSphere Application Server Enterprise features such as the scheduler, extended messaging and Process Choreographer.

Things to check

1. You can only involve a single one-phase commit resource or you will receive an error.
2. Verify that last participant support is enabled on your application, either in the .ear file or configured after install.
3. If heuristic reporting is enabled, review the reports to check on the one-phase commit status.

Symptom: Transaction rolls back

Look in the JVM logs for this message.

Example 8-16 One-phase commit resource tries to participate with two-phase

CHANGE THIS ERROR MESSAGE!

WTRN0063E: An illegal attempt to commit a one phase capable resource with existing two phase capable resources has occurred.

You are trying to use more than one one-phase commit resource. In this transaction, only a single one-phase commit resource can be included.

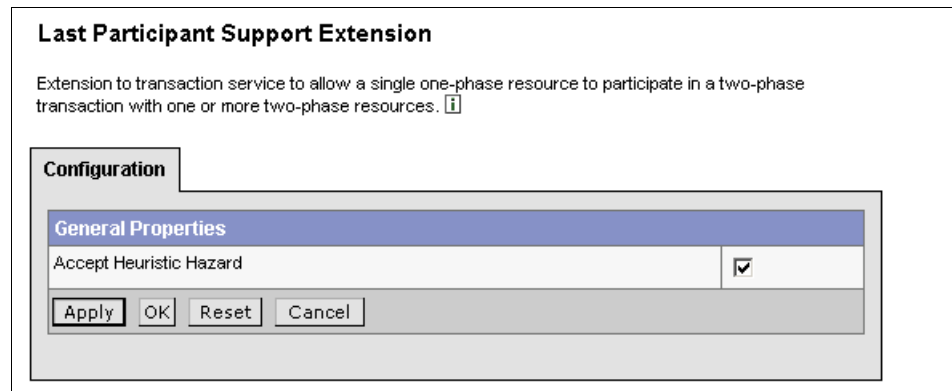


Figure 8-3 Last participant support option on an application in the Administrative Console

Symptom: WTRN0061W: A heuristic condition may have occurred for transaction 00000010214b4786....

One or more two-phase commit transactions failed to commit. The one-phase commit cannot be rolled back and there is a mixed state. If the server restarted during a transaction, the status of the one-phase commit transaction may be unknown. The two-phase commit transaction will then be rolled back.

Try enabling heuristic reporting to keep track of one-phase commit transactions.
To enable reporting:

1. In the Administrative Console, go to **Servers-> Manage Application Servers-> <server_name>**.
2. Select **Transaction Service**.
3. On the Configuration tab, enter a log directory, check **enableLoggingForHeuristicReporting**, click **OK** and save.
4. Restart the server.

8.1.13 ActivitySession service

The ActivitySession service provides an alternate unit-of-work (UOW) scope to the scope provided by global transaction contexts. An ActivitySession context can be longer-lived than a global transaction context and can encapsulate global transactions.

Things to check

1. Review the logs for error messages. The ActivitySession service produces diagnostic messages prefixed by WACS. Activity log messages produced by the ActivitySession service are accompanied by Log Analyzer descriptions.
2. Do not nest activity sessions.
3. If you are on V5.0 or V5.0.1, review the WebSphere Application server support site for ActivitySession related APARs.

Symptom: Activity session not working, com.ibm.websphere.ActivitySession.NotSupportedException

You would see this error if you have a nested activity session which is not supported.

8.1.14 Back-up cluster support

Back-up cluster support allows you to failover a cluster with EJBs installed to a cluster in another cell. Fallback happens automatically.

In the eMerge scenario, EJBs involved in the business choreographer could be installed on a cluster with a mirrored cluster available as back-up.

Things to check

1. Check that your back-up cluster has the same application, same application name, and same resources as the primary cluster.
2. The primary and backup cluster need to be back-up clusters for each other to have fallback. All the servers and deployment managers involved must be at WebSphere Application Server Enterprise or WebSphere Network Deployment 5.0.2 or later. Back-up cluster support is only available starting with 5.0.2.
3. Check that the bootstrap address of your back-up cluster is correct for the Deployment Manager of this cluster.
4. Ensure that your back-up server is running and the application is configured correctly.
5. Check that your systems can see each other on the network.

Symptom: Cluster does not failover

Review the JVM logs on both systems. Did a failover occur, but the back-up cluster shows errors with the application? Check that your back-up cluster is running and that the application started without errors in the JVM log.

Check that all of the primary servers are down. Failover only takes place when all of the primary servers are down.

Make sure that you are not trying to failover Web modules. You cannot failover a Web module. You may have to split your application's Web and EJB modules over two clusters to failover the EJB modules. Watch for JNDI problems if you split your application across clusters. You may have to modify your application to preface your original JNDI name with the full name to the cluster, for example, `cell/clusters/eMergeCluster/<rest_of_JNDI_name>`. See the WebSphere InfoCenter article on JNDI interoperability considerations.

Verify that all the installs involved with the clusters are at 5.0.2.

Check that you have the correct bootstrap host assigned for the back-up cluster. This should be the bootstrap address of the back-up cluster's Deployment Manager.

The primary cluster and back-up cluster must be in separate cells.

Symptom: Failover occurred, but application does not work

Review the JVM logs on both systems. Check that your back-up cluster is running and that the application started without errors in the JVM log.

Make sure that you are not trying to failover Web modules. You cannot failover Web modules. You may have to split your application's Web and EJB modules over two clusters to failover the EJB modules. Watch for JNDI problems if you split your application across clusters. You may have to modify your application to preface your original JNDI name with the full name to the cluster. For example, `cell/clusters/eMergeCluster/<rest_of_JNDI_name>`.

If your whole cell has died (Deployment Manager and nodeagent), then new connections cannot be routed to the back-up cluster.

Symptom: Fallback did not happen

Check that there are primary cluster members started. Fallback occurs when any of the primary cluster members are available.

Check that your primary cluster is configured as a back-up of the back-up cluster

The client connecting to the cluster may have made new connections only to the back-up cluster. The new connections do not know about the primary cluster. Connections that existed prior to failover should fail back.

8.1.15 Administrative Console (Enterprise)

When WebSphere Application Server Enterprise is installed, additional menus and options are added to the Administrative Console to work with WebSphere Application Server Enterprise features.

If you experience problems using the Administrative Console, refer to the WebSphere Application Server and WebSphere Network Deployment sections on the Administrative Console.

If WebSphere Application Server Enterprise resources or options are missing, review `PMEinstallSummary.log` and other WebSphere Application Server Enterprise install logs to ensure that those features were installed.

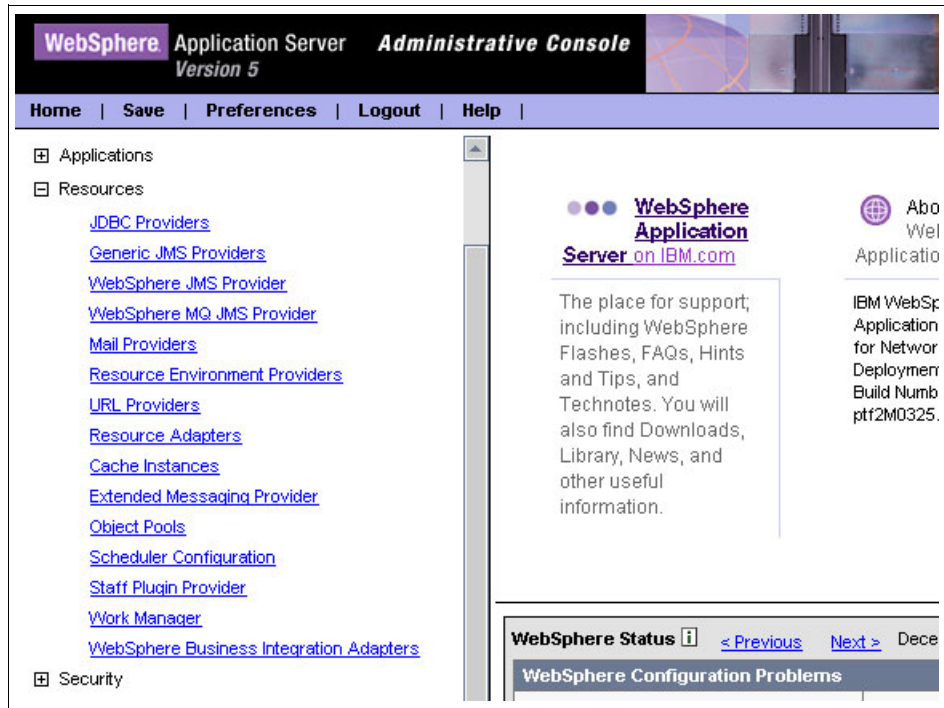



Figure 8-4 Administrative Console with all WebSphere Application Server Enterprise resources installed



WebSphere Application Server Network Deployment

In a simple environment, one or two WebSphere Application Servers can easily be managed individually; they can even share resources between them. In real life, finding only one or two application servers in a solution is very unlikely.

In a complex environment where several application servers are running either separately or in a cluster, the complexity of management increases significantly. Managing servers individually is not acceptable. Sharing resources and managing clusters without a centralized system is not an option.

WebSphere Application Server Network Deployment provides quality of service in a complex application server environment, including scalability, high availability, manageability through providing centralized management, clustering, resource sharing, etc.

This chapter focuses on the most common problems in a WebSphere Application Server Network Deployment environment.

9.1 WebSphere Application Server Network Deployment environment

In this section, we will discuss the the problems in a cell environment along with Deployment Manager, which is included in WebSphere Application Server Network Deployment and WebSphere Application Server Enterprise. First, let us clarify some concepts.

- ▶ Managed process

A Managed process is an individual server or a process, like the WebSphere Application Server or JMS Server.

- ▶ Node

A Node is a logical grouping of Managed processes, managed via a nodeagent. A Node usually corresponds to a physical computer system with a distinct IP host address. Node names usually are identical to the host name for the computer.

- ▶ Cluster

A Cluster is a logical grouping of application servers which can be managed as a unit and participate in workload management. Servers that belong to a Cluster are members of that cluster set and must all have identical application components deployed on them. Other than the applications configured to run on them, cluster members do not have to share any other configuration data. The cluster members of a cluster can be on different host machines.

- ▶ Cell

A Cell is an aggregation of Nodes in a WebSphere Application Server distributed network. A Deployment Manager on the Cell controls and communicates with all the nodeagents. Deployment Manager allows you to manage multiple WebSphere Application Server nodes from a single, central location. You can install Deployment Manager on any machine in the network to create a cell.

When the single server cannot satisfy the customer's requirement, for example, the workload increases rapidly and workload management becomes necessary; in this case, a WebSphere Application Server distributed network may be considered.

9.1.1 Deployment Manager

In the WebSphere Application Server distributed network environment (cell environment), Deployment Manager is an important component.

Deployment Managers are administrative agents that provide a centralized management view for all nodes in a cell, as well as management of clusters and workload balancing of application servers across one or several nodes. A Deployment Manager does not host any customer applications, but hosts the Administrative Console. A Deployment Manager provides a single, central point of administrative control for all elements of the entire WebSphere Application Server distributed cell.

Working with base servers

WebSphere Application Server nodes can be easily federated into one Deployment Manager cell.

A coexistence environment might have multiple WebSphere Application Server product installations on one machine. You can federate each installation into the same cell, or into different cells. Whenever the Deployment Manager federates a node into its cell, it configures the nodeagent server process for the node with a set of default ports. If the base node has the embedded messaging feature, the deployment manager also configures a WebSphere Application Server JMS Provider, jmsserver, which is a server process with another set of default ports.

In contrast to coexistence, there are multiple configuration instances; this refers to multiple instances sharing the same product installation. The multiple configuration instances cannot be federated into the cell.

Deployment Manager communicates with the nodeagents in the cell, and the nodeagent manages all of the servers in the node, including the application servers and JMS server. Figure 9-1 on page 214 provides a simple illustration.

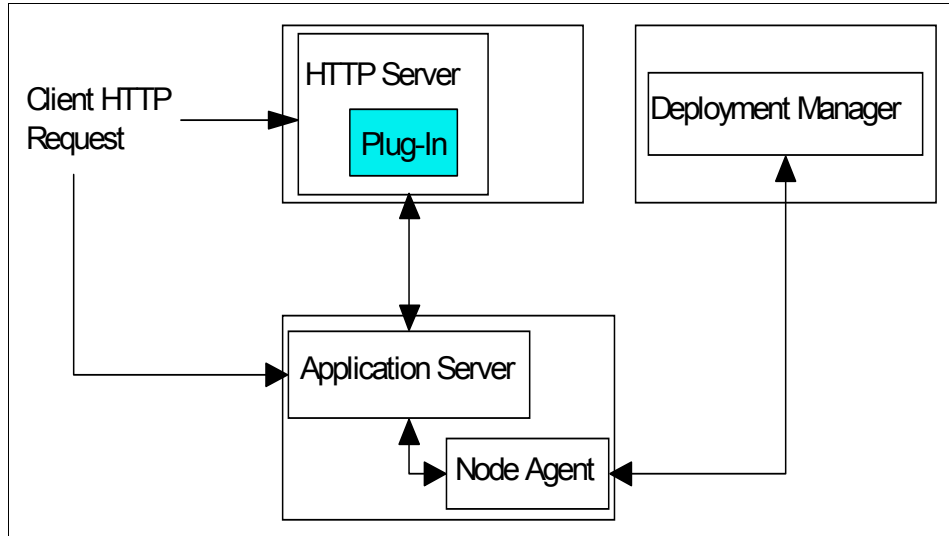


Figure 9-1 Deployment Manager working with application servers

Working with Enterprise servers

The Enterprise installation image contains programming model extensions to the core Application Server, such as Business Rule Beans and Process Choreographer, and also contains Deployment Manager extensions for administering functions included in the programming model extensions.

Although the Enterprise package includes a separate base WebSphere Application Server product CD-ROM, the Enterprise product supports an umbrella installation, which automatically installs the base product within the same installation procedure when installing the Enterprise product on a clean machine. If you install the Enterprise product on a machine with an installed base Application Server product, the Enterprise product checks to see if there are any required base features that are not already installed. If so, the Enterprise product installs the features, and extends the base Application Server product with Enterprise extensions.

On the other hand, although the Enterprise package includes the Network Deployment product CD-ROM, installing the Enterprise product does not install the Network Deployment product. You must use the Network Deployment product CD-ROM to install the Network Deployment product. If you install the Enterprise product on a machine with an installed Network Deployment product, the Enterprise product extends the Network Deployment Administrative Console.

Figure 9-2 on page 215 gives a simple illustration of a Network Deployment cell environment with Enterprise extensions.

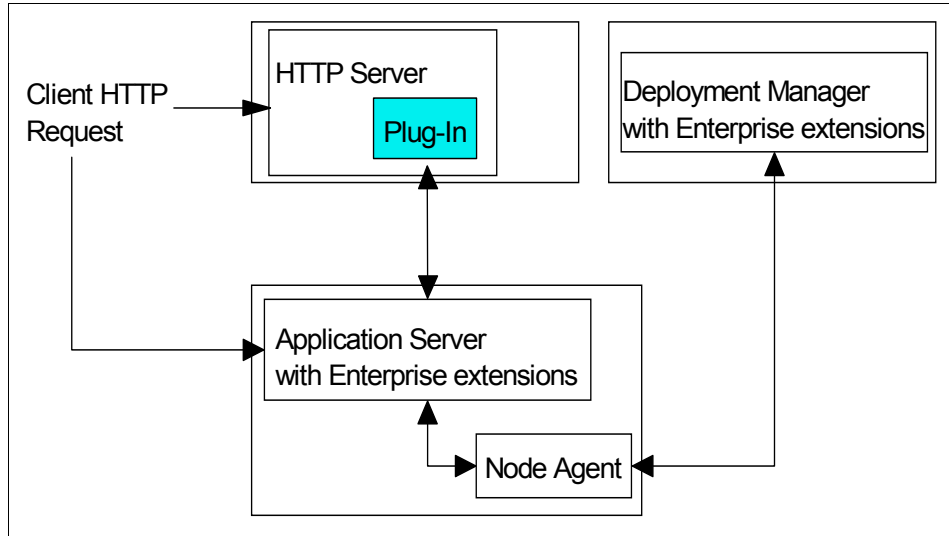


Figure 9-2 Deployment Manager working with enterprise servers

Note: If you want to use Enterprise extensions in one cell, you must install the Enterprise product on the deployment manager node and on all base product nodes that are to become part of the cell.

9.1.2 Problems in the cell environment

In the following section, we will discuss some common problems in cell environment.

Node federation

Although it is a configuration problem instead of a runtime problem, we would like to discuss the problem that arises when federating nodes into a cell, because it is very common.

There are two ways to federate nodes into the cell:

- ▶ Using the deployment manager Administrative Console.
- ▶ Using the **addNode** command line script from the bin directory of the node you are federating; this is **addNode.sh** on UNIX.

Things to check

If you encounter problems when federating a node, the following items must be checked first.

1. Do the node and Deployment Manager have the same product extensions?
For example, is the Deployment Manager upgraded with the Enterprise extensions if the node is an Enterprise server, or vice versa?
You may use **versionInfo.sh** to verify the product extensions; for information about the usage of the **versionInfo** command, please refer to 4.3, “WebSphere Application Server” on page 52.
2. Does the product level of the WebSphere Application Server installation match that of the Deployment Manager?
For example, is the Fix Pack 2 of Application Server applied if that of Deployment Manager is applied?
You may also use **versionInfo.sh** to verify the product level.
3. Does the network work properly?
For example, can you ping the Deployment Manager host from the federating node?
4. Are you using the right port? If you do not know which port this is, please check in the Administrative Console.
For example, the SOAP connector port will be used if you do not specify the **-conntype** option when federating the node using the **addNode** command; you may find the port number in the Administrative Console of the Deployment Manager by clicking **System Administration -> Deployment Manager -> End Points -> SOAP_CONNECTOR_ADDRESS**.
If you use the Administrative Console of the Deployment Manager to federate another node, you may find the SOAP port number in the Administrative Console of that server by clicking **Servers -> Application Servers -> <server_name> -> End Points -> SOAP_CONNECTOR_ADDRESS**.
5. Is the security of the cell enabled? If so, do you provide **-username** and **-password** options when federating the node?
6. Check that the domain name resolutions for the Deployment Manager and federating node are correct on both machines. In other words, is the mapping of the IP address to the host name correct? Incorrect resolution might lead into potential issues.
7. Check the system times on the node and Deployment Manager; if these are too far apart, you cannot add a node. Modify the times so they are synchronized.
8. Review the **addNode.log** or use the **-trace** option with the **addNode** command.

Symptom: Cannot addNode

As mentioned previously, you can federate the node through the deployment manager Administrative Console or using the **addNode** command.

If you federate the node through the **addNode** command, there will be some brief information shown on the screen, as in Example 9-1.

Example 9-1 addNode by command

```
# ./addNode.sh 9.24.104.190 8889
ADMU0116I: Tool information is being logged in file
           /usr/WebSphere/AppServer/logs/addNode.log
ADMU0001I: Begin federation of node m1097504 with Deployment Manager at
           9.24.104.190:8889.
ADMU0124E: The system clock of the new node is not synchronized with that of
           the deployment manager.
ADMU0125I: Change the clock of the new node to be within 5 minutes of the clock
           of the deployment manager.
```

In most cases, you will find the reason of the **addNode** command failure in the screen output. For example, the reason in the above example is The system clock of the new node is not synchronized with that of the deployment manager, and the user response should be Change the clock of the new node to be within 5 minutes of the clock of the deployment manager.

If you think the explanation in the screen is not enough, you may find more detailed information in the Messages Reference of the InfoCenter, at:

<http://publib.boulder.ibm.com/infocenter/wasinfo/index.jsp>

. For example, if you want to know more about the message identifier ADMU0124E, you may click **Quick reference -> Messages -> ADMU** in the Contents area of InfoCenter, and find ADMU0124E on the right page, as shown in Figure 9-3 on page 218.

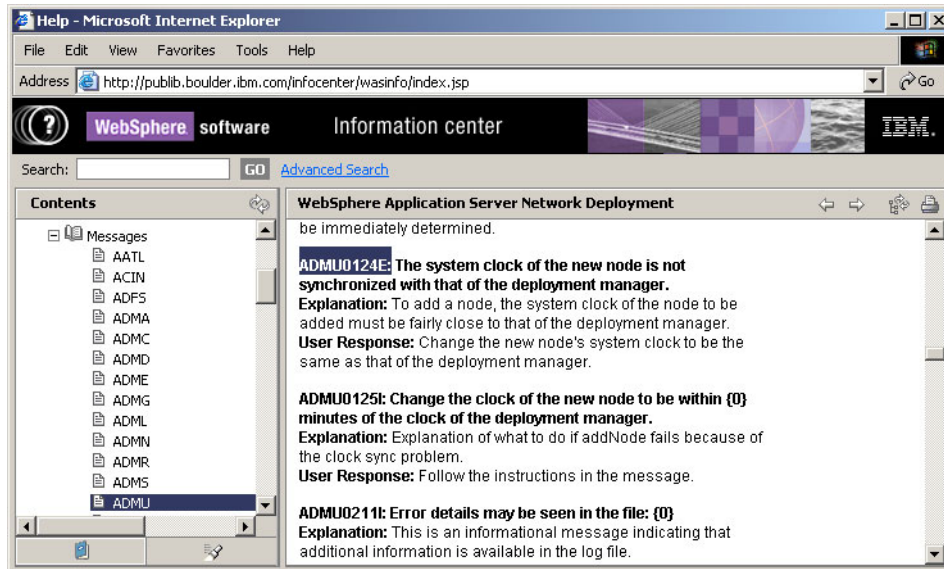


Figure 9-3 Messages Reference of Information Center

If you think the information provided on the screen is too sparse, you may review the `addNode.log`, the location of which is indicated in the output of the command. For example, the default location is `<WebSphere_root>/logs/addNode.log` and the content is as shown in Example 9-2.

Example 9-2 addNode.log

```
***** Start Display Current Environment *****
Host Operating System is AIX, version 5.1
Java version = J2RE 1.3.1 IBM AIX build ca131-20030329 (JIT enabled: jitc),
Java Compiler = jitc, Java VM name = Classic VM
was.install.root = /usr/WebSphere/AppServer
user.install.root = null
Java Home = /usr/WebSphere/AppServer/java/jre
ws.ext.dirs = ... ..
Classpath = ... ..
Java Library path = ... ..
Current trace specification = com.ibm.ws.management.tools.*=all=enabled
***** End Display Current Environment *****
[11/21/03 17:35:33:238 EST] 4138b3 ManagerAdmin I TRAS0017I: The startup
trace state is com.ibm.ws.management.tools.*=all=enabled.
[11/21/03 17:35:33:726 EST] 4138b3 AdminTool d executing utility with
arguments: /usr/WebSphere/AppServer/config m1097504 m1097504 9.24.104.190 8889
[11/21/03 17:35:59:750 EST] 4138b3 AdminTool A ADMU0001I: Begin
federation of node m1097504 with Deployment Manager at 9.24.104.190:8889.
```

```

[11/21/03 17:36:11:205 EST] 4138b3 AbstractNodeC d DeploymentManager handle
is ==>
WebSphere:cell=ka0k1tbNetwork,mbeanIdentifier=DeploymentManager,name=Deployment
Manager,node=ka0k1tbManager,platform=common,process=dmgr,type=DeploymentManager
,version=5.0
[11/21/03 17:36:14:241 EST] 4138b3 AbstractNodeC d Server handle is ==>
WebSphere:cell=ka0k1tbNetwork,mbeanIdentifier=cells/ka0k1tbNetwork/nodes/ka0k1t
bManager/servers/dmgr/server.xml#Server_1,name=dmgr,node=ka0k1tbManager,platfor
m=common,process=dmgr,processType=DeploymentManager,type=Server,version=5.0
[11/21/03 17:36:14:801 EST] 4138b3 AbstractNodeC d JVM handle is ==>
WebSphere:cell=ka0k1tbNetwork,mbeanIdentifier=JVM,name=JVM,node=ka0k1tbManager,
platform=common,process=dmgr,type=JVM,version=5.0.1
[11/21/03 17:36:14:859 EST] 4138b3 AdminTool A ADMU0124E: The system
clock of the new node is not synchronized with that of the deployment manager.
[11/21/03 17:36:14:891 EST] 4138b3 AdminTool A ADMU0125I: Change the
clock of the new node to be within 5 minutes of the clock of the deployment
manager.

```

If the information in the `addNode.log` is still not enough to determine the problem, you may run the `addNode` command with the `-trace` option again, and the trace information will also be logged into the `addNode.log`, as shown in Example 9-3 and Example 9-4.

Example 9-3 addNode by command with -trace option

```

# ./addNode.sh 9.24.104.190 8889 -trace
ADMU0115I: Trace mode is on.
ADMU0116I: Tool information is being logged in file
        /usr/WebSphere/AppServer/logs/addNode.log
ADMU0001I: Begin federation of node m1097504 with Deployment Manager at
        9.24.104.190:8889.
ADMU0124E: The system clock of the new node is not synchronized with that of
        the deployment manager.
ADMU0125I: Change the clock of the new node to be within 5 minutes of the clock
        of the deployment manager.

```

Example 9-4 addNode.log when trace mode is on

```

***** Start Display Current Environment *****
Host Operating System is AIX, version 5.1
Java version = J2RE 1.3.1 IBM AIX build ca131-20030329 (JIT enabled: jitc),
Java Compiler = jitc, Java VM name = Classic VM
... ..
Current trace specification = com.ibm.*=all=enabled
***** End Display Current Environment *****
[11/24/03 13:55:47:239 EST] 47183a ManagerAdmin I TRAS0017I: The startup
trace state is com.ibm.*=all=enabled.

```

```

[11/24/03 13:55:47:746 EST] 47183a AdminTool d executing utility with
arguments: /usr/WebSphere/AppServer/config m1097504 m1097504 9.24.104.190 8889
-trace
[11/24/03 13:55:56:566 EST] 47183a AdminClientFa > createAdminClient
[11/24/03 13:55:56:598 EST] 47183a AdminClientFa > loadPropertiesFromFile

file:/usr/WebSphere/AppServer/properties/soap.client.props
[11/24/03 13:55:56:877 EST] 47183a AdminClientFa < loadPropertisFromFile
[11/24/03 13:55:57:192 EST] 47183a AdminClientFa d [key, value]
                                cellName
                                m1097504
[11/24/03 13:55:57:521 EST] 47183a AdminClientFa d [key, value]
                                port
                                8889

... ..
[11/24/03 13:56:30:971 EST] 47183a GenericSerial > replaceObject
                                websphere.addnode.message
                                ADMU0124E: The system clock of the new node is
not synchronized with that of the deployment manager.
                                {nodeName=m1097504}
[11/24/03 13:56:30:971 EST] 47183a GenericSerial < replaceObject
                                websphere.addnode.message
                                ADMU0124E: The system clock of the new node is
not synchronized with that of the deployment manager.
                                {nodeName=m1097504}

... ..

```

For information on how to read the trace, please refer to 4.3, “WebSphere Application Server” on page 52.

In general, customers do not need to be familiar with how to read the trace; you may contact the IBM support team to ask for further help, but you should know how to obtain the trace because the IBM support team may ask you to provide it to diagnose the problem.

A possible reason for an **addNode** command failure is that the deployment manager Domain Name Server (DNS) configuration is set up improperly. For example, the default installation uses the loopback address (127.0.0.1) as the default host address. If file transfer traces at the node show the node trying to upload files to a URL that includes 127.0.0.1, the node has an incorrect DNS configuration. To correct this problem, update the `/etc/hosts` file or the name service configuration file, `/etc/nsswitch.conf`, to query the Domain Name Server or Network Information Server (NIS) before searching hosts.

If you federate the node through the deployment manager Administrative Console, there is a more condensed information in the console, as shown in Figure 9-4 on page 221.

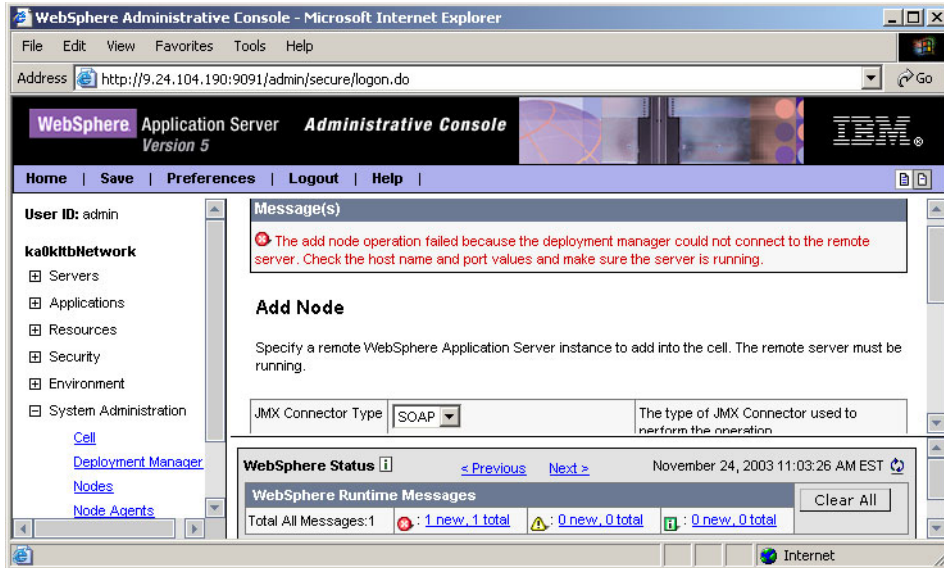


Figure 9-4 Adding node in Deployment Manager Administrative Console

Concurrently, please pay attention to the number of error messages in the WebSphere Runtime Messages area. If the number is not equal to zero, you may click the number of messages to view the message list, as shown in Figure 9-5.

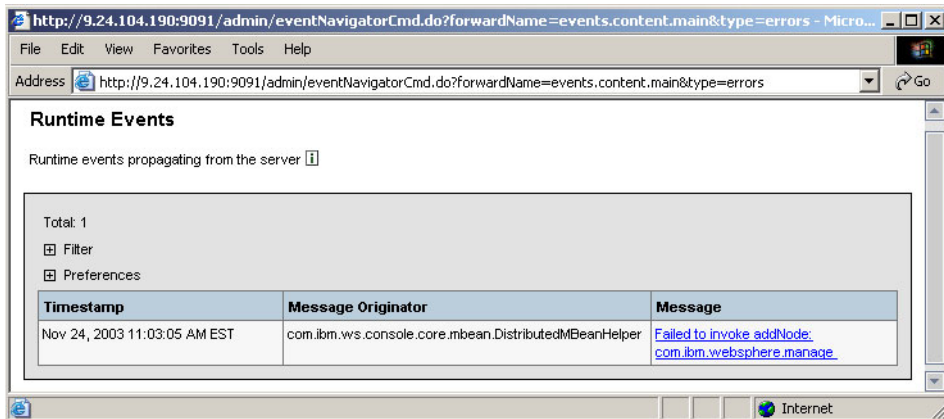


Figure 9-5 Error messages list

You may then click the specific message to view the details, as shown in Figure 9-6 on page 222.

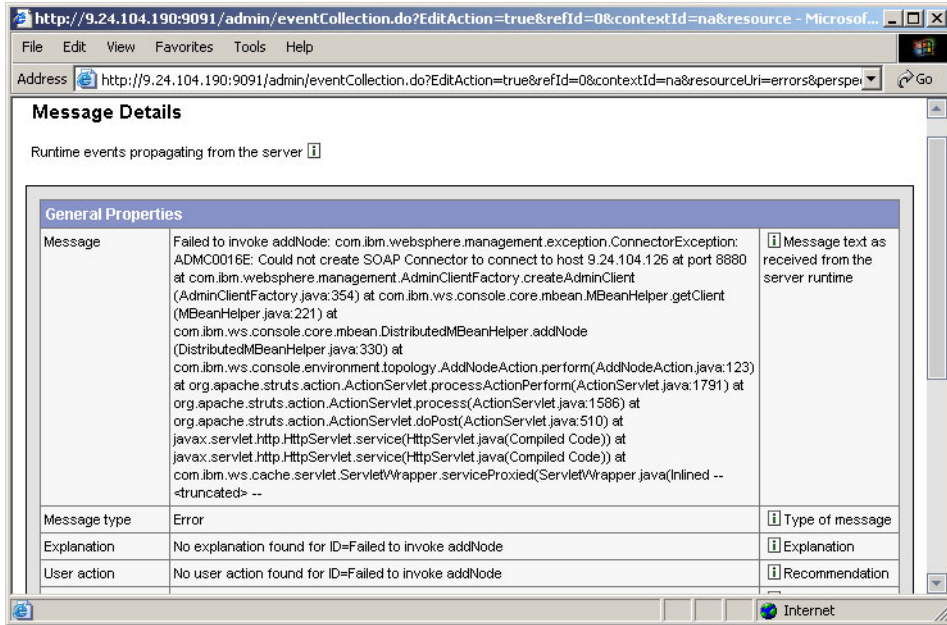


Figure 9-6 One specific error message

If the problem is ADMC0016E: Could not create SOAP Connector to connect to host {0} at port {1}, you may find the explanation in the InfoCenter:

Explanation: In most cases, the wrong host and port are provided. If the remote process runs in secure mode, appropriate user and password are required.

In this case, you must verify that the IP address is correct when you provide it to the Host field, that the hostname can be resolved to the correct IP address when you provide it to the Host field, that the port number is correct, and so on.

If the problem occurs after obtaining the connection, addNode.log will be generated on the federating node. You may then get further information in the addNode.log.

There are also two ways to unfederate nodes from the cell:

- ▶ Using the deployment manager Administrative Console.
- ▶ Using the **removeNode** command line script from the bin directory of the node you are unfederating.

The problems of unfederating nodes are fewer than those of federating nodes. The problem determination process is similar as for federating nodes. The main

difference is that the log name is `removeNode.log` instead of `addNode.log`. We will not discuss this again.

Symptom: Applications are lost

After you federate the node, you may find that the applications in the node are lost; where are they?

If you federate the node using the `addNode` command, make sure you specify the `-includeapps` option. If you do not specify it, the applications in the node will not be uploaded into the Network Deployment environment. You may reinstall the applications, or remove the node and add it again using the `-includeapps` option.

Similarly, if you federate the node in the Deployment Manager Administrative Console, please make sure you check the **Include Applications** checkbox.

If the `-includeapps` option is specified or checked, please review the `addNode.log`; you can also enable the trace to get further information, as mentioned in “Node federation” on page 215.

9.1.3 Administrative Console

Using the Administrative Console with the Deployment Manager is similar to using it on a stand-alone WebSphere Application Server install, but you can administer multiple servers, nodes, and clusters.

Review the Administrative Console section in WebSphere Application Server for information on basic problem determination using the Administrative Console. This section will cover additional problems related to the Administrative Console on a Deployment Manager.

Things to check

1. Check that the deployment manager is running.
2. Verify that the nodes are synchronized.
3. Review JVM logs for Deployment Manager for Administrative Console messages.
4. If you stop the deployment manager from the Administrative Console, open a new browser to log in the next time.
5. For problems with Console users, refer to Console users in the WebSphere Application Server section.
6. Enable tracing:
 - a. Stop the Deployment Manager.

- b. Edit the server.xml file under the following directory:
<WebSphere_root>/config/cells/<cell_name>/nodes/<node_name>/servers/dmgr.
- c. Search for startupTraceSpecification and change the string after it from
=all=disabled to com.ibm.ejs.=all=enabled.
- d. Save the file server.xml with the above changes and stop and restart the Deployment Manager.
- e. Reproduce the problem by bringing up the console in your browser and check the trace.log for any trace information and errors.

Symptom: Nodes not synchronized

If the nodes are not synchronized, you may see a different behavior if changes were made to the Deployment Manager repository, but not pushed out to all the nodes. Review the JVM logs on the deployment manager and the unsynchronized node to look for error messages about why the node is not synchronized. There may be a security, network or system time problem.

You can also try running the **syncNode** command if synchronizing the node from the Administrative Console does not work.

Symptom: ADMN0022E: Access denied for...MBean due to insufficient or empty credentials

If you see this error message, make sure that your node is synchronized. You may need to run the **syncNode** command on the node to ensure that the security information is correctly pushed to the node.

Symptom: Status on a process is unknown

Use the **Refresh** button on the status column to get a new status. If the status is still unknown, check on the status of the nodeagent in the Administrative Console. If it is unknown, check the nodeagent machine. Start or restart the nodeagent. Review the JVM logs for errors on the nodeagent if it was running but not showing up in the Administrative Console, or stopped running on its own.

Symptom: Cannot open Administrative Console

Verify that the deployment manager is running. Review the deployment manager JVM to see if there was any problem starting the Administrative Console application. Make sure that there is no firewall preventing access to the machine.

Symptom: Used removeNode, but applications are still appear in the Administrative Console

If you used the `-includeapps` option on an `addNode`, those applications will not be uninstalled on a `removeNode` since they may be used by other nodes. If you want to remove them, you must uninstall them separately.

9.1.4 Workload management

Workload management optimizes the distribution of client processing requests. Incoming work requests are distributed to the application servers, enterprise beans, servlets, and other objects that can most effectively process the requests. Workload management also provides failover when servers are not available, improving application availability.

In the WebSphere Application Server environment, workload management is implemented by using clusters of application servers.

There are three types of requests that can be workload managed in WebSphere Application Server V5.0.

1. HTTP requests can be shared across multiple HTTP Servers.
2. Servlet requests can be shared across multiple Web containers.
3. EJB requests can be shared across multiple EJB containers.

We will discuss the three types briefly.

Workload management of HTTP requests

HTTP requests can be shared across multiple HTTP Servers, as shown in Figure 9-7 on page 226.

- ▶ This requires a TCP/IP sprayer to take the incoming requests and distribute them. There are hardware and software products available to spray TCP/IP requests.
- ▶ The Load Balancer (formerly Network Dispatcher) component of the Edge Components (formerly WebSphere Edge Server) is a software solution that applies intelligent load balancing to HTTP requests.

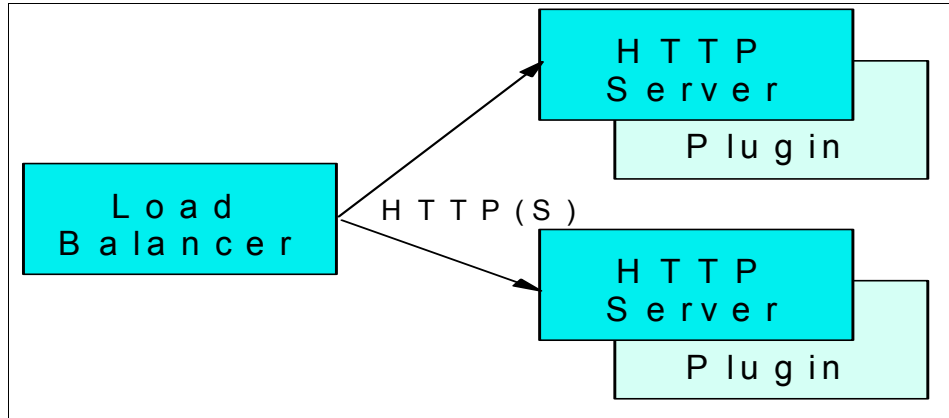


Figure 9-7 Workload management of HTTP requests

For problems about workload management of HTTP requests, please refer to the specific product documentation.

We will not discuss Load Balancer in this book. If you encounter problems with Load Balancer, please refer to the chapter “Administering and troubleshooting Load Balancer” of the *Load Balancer Administration Guide* at:

<http://www-306.ibm.com/software/webservers/appserv/doc/v50/ec/infocenter/>

Workload management of servlet requests

Servlet requests can be shared across multiple Web containers, as shown in Figure 9-8 on page 227.

- ▶ The WebSphere Application Server plug-in to the HTTP Server distributes servlet requests.
- ▶ Web containers can be configured on the same machine or multiple machines.

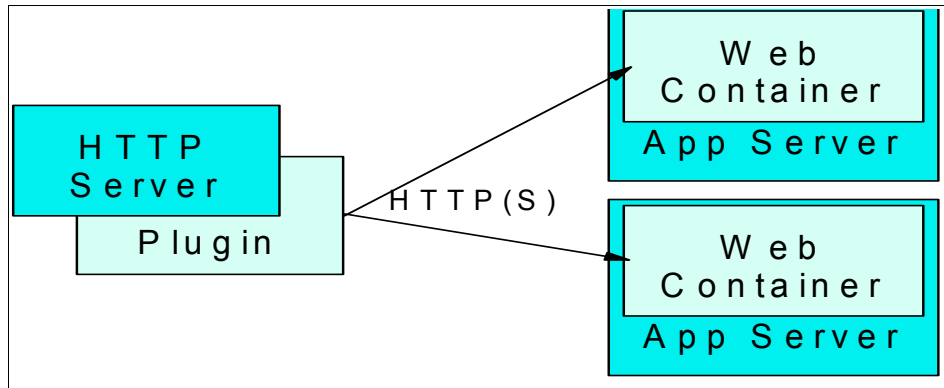


Figure 9-8 Workload management of Servlet requests

If HTTP requests are not being distributed to all servers, you should check the following:

- ▶ Ensure that all the machines in your configuration have TCP/IP connectivity to each other by running the **ping** command:
 - From the HTTP Server to each WebSphere Application Server.
 - From each WebSphere Application Server to the HTTP Server.

- ▶ Check the PrimaryServers list in the plugin-cfg.xml.

The plug-in load balances across all servers defined in the PrimaryServers list, if affinity has not been established. If you do not have a PrimaryServers list defined, the plug-in load balances across all servers defined in the cluster, if affinity has not been established. In the case where affinity has been established, the plug-in should go directly to that server for all requests within the same HTTP session.

- ▶ If some servers are servicing requests and one or more others are not, try accessing a problem server directly to verify that it works, apart from workload management issues.
 - If it does not work:
 - Make sure that the affected server is running. You may use the Administrative Console to do so.
 - If the affected server is running but still does not work, please refer to Chapter 7, “WebSphere Application Server: base” on page 135 for further help.
 - If it works, please refer to Chapter 6, “WebSphere Application Server : plug-in” on page 123 for further help.

Workload management of EJB requests

EJB requests can be shared across multiple EJB containers, as shown in Figure 9-9.

- ▶ The Workload Management plug-in to the Object Request Broker (ORB) distributes EJB requests.
- ▶ EJB requests can come from servlets, Java client applications, or other EJBs.

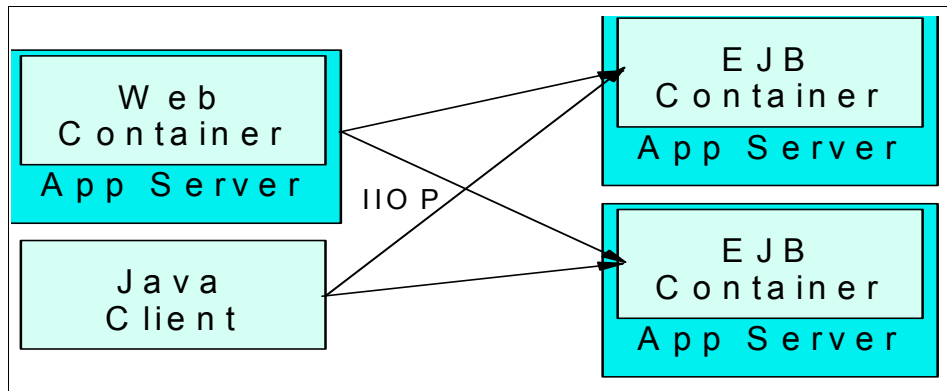


Figure 9-9 Workload management of EJB requests

In this chapter, we will put emphasis on the workload management of EJB requests.

Things to check

When you encounter a problem relating to workload management of EJB requests, you should eliminate environment or configuration issues first.

1. Make sure the nodes were federated successfully.
There is a log file named `addNode.log` on each federated node. If the node was federated successfully, you may find the following line at the end of the log file:
`AdminTool A ADMU0003I: Node ka0k1tb has been successfully federated.`
2. Make sure the network works normally.
Please make sure that the network connection is working between Deployment Manager and each node, and between the EJB client and the EJB server.
3. Make sure the application servers are in the status you want; for example, all of the members of the cluster are running. You may verify this in the Administrative Console.

4. Make sure the application which you think is failing is installed and running.
You also may verify the running status and synchronization status in the Administrative Console. On the other hand, you may verify the physical files on each node.
5. Make sure the weight of each cluster member is set to the allowed values.
You may verify the weight in the Administrative Console, by clicking **Servers -> Clusters -> <cluster_name> -> Cluster members -> <member_name>**, as show in Figure 9-10 .

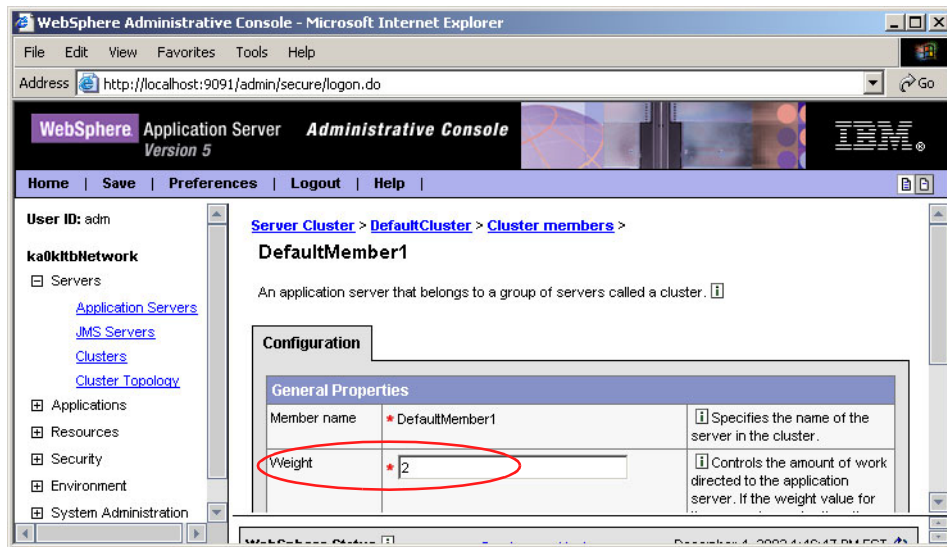


Figure 9-10 Check the weight of cluster members

Ensure that the weights are within the valid range of 0-20. If a server has a weight of 0, no requests will be routed to it. Weights greater than 20 are treated as 0.

6. If the problem relates to distributing the workload across persistent (CMP or BMP) enterprise beans, make sure the supporting JDBC drivers and datasources are available on each server.

Symptom: EJB requests are not distributed to all servers

If a client cannot reach a server in a cluster thought to be reachable, a server may be marked unusable, or may be down.

Follow the steps in “Things to check” on page 228 of this chapter first.

If possible, try accessing the enterprise bean directly on the problem server to see if there is a problem with TCP/IP connectivity, application server health, or other issues not related to workload management.

If this fails, please refer to Chapter 7, “WebSphere Application Server: base” on page 135 for further help. If it works, please refer to “Get more information” on page 231.

Symptom: EJB requests are not distributed evenly

There are a number of possible reasons for this behavior, which generally fall into one or more of the following categories:

- ▶ Improper configuration.
- ▶ Environment issues, such as the availability of servers or applications.
- ▶ A large numbers of requests that involve transactional affinity.
- ▶ A small number of clients.

Workload management of WebSphere Application Server is based on a round robin scheme of request distribution. This results in the balance being determined by the numbers of requests rather than by any other measure. A true balance problem is determined by comparing the number of requests processed by each member of the cluster with the weights that have been set for each of those members.

- ▶ When the percentage of requests that arrive for each member of the cluster is consistent with the weights then further analysis of the application is required to determine the cause for the workload being imbalanced even when the number of requests is balanced.
- ▶ When the number of *numIncomingNonWLMObjectRequests* is not balanced among the members of the cluster and is large in relation to the *numIncomingRequests* then the reason for the imbalance is the non-distributable components installed on the members of the cluster. A modification to the configuration will yield a more balanced environment.
- ▶ When the number of *numIncomingStrongAffinityRequests* is not balanced among the members of the cluster and is large in relation to the *numIncomingRequests* then the reason for the imbalance is the requests invoked within a transaction. These can be reduced by installing the objects involved in a transaction within the same cluster.

The *numIncomingNonWLMObjectRequests*, *numIncomingStrongAffinityRequests* and *numIncomingRequests* are the counters of Performance Monitoring Service. You can monitor them using Tivoli Performance Viewer.

Symptom: Failover fails

This problem means that a failing server still receives enterprise bean requests.

The client might have been in the middle of a transaction with an Enterprise bean on the server that went down. If a request is returned with CORBA

`SystemException COMM_FAILURE`

`org.omg.CORBA.completion_status.COMPLETED_MAYBE`, this might be working as designed. The design is to let this particular exception flow back to the client, since the transaction might have completed. Failing over this request to another server could result in this request being serviced twice.

If the requests sent to the servers come back to the client with any other exceptions in a consistent manner, it may be that no servers are available.

Symptom: Servers do not share the workload after being restored

This error occurs when the servers that were unavailable are not recognized by the Workload Management component after they are restored. There is an unusable interval determined by the property `com.ibm.websphere.wlm.unusable.interval` during which the Workload Manager waits to send to a server that has been marked unusable.

If you want to change this, you may specify the following command-line arguments in the Generic JVM arguments field on the Java Virtual Machine page:

```
-Dcom.ibm.websphere.wlm.unusable.interval=interval
```

The *interval* is the time in seconds.

Get more information

If the problems are not solved, or you encounter other problems with EJB workload management, please note that you can get more information using the following sources.

1. The `activity.log` is a good start.

You may read the `activity.log` using Log Analyzer; you may even merge all the `activity.logs` of all of the cluster members. Check the `activity.log` for entries that show any of the following:

- A server has been marked unusable more than once and remains unusable.
- All servers in a cluster have been marked unusable and remain unusable.

- A Location Service Daemon (LSD) has been marked unusable more than once and remains unusable.

Look for the following entries:

- WWLM0061W: An error was encountered sending a request to cluster member {0} and that member has been marked unusable for future requests to cluster {1}, because of exception {2}

Explanation: A cluster member has been marked unusable for normal operational reasons (a ripple start).

User Response: No user action is required. However, if the problem persists, check the server status (stopped/started/running, etc.). If the cluster member is down, restart and continue. If the server is still running, check that it is reachable across the network, and then check that the weights assigned to the server are in the accepted range. If the weight assigned is valid, check the server logs for possible errors.

- WWLM0062W: An error was encountered sending a request to cluster member {0} and that member has been marked unusable, for future requests to cluster {1}, two or more times, because of exception {2}

Explanation: A cluster member which was thought to be reachable has been marked unusable two or more times.

User Response: Check the cluster member status (stopped/start/running, etc.). If the cluster member is down, restart and continue. If the cluster member is running, check that it is reachable across the network, and then check that the weights assigned to the cluster member are in the accepted range. If the weight assigned is valid, check the cluster members logs for possible errors.

- WWLM0063W: An error was encountered attempting to use the Location Service Daemon {0} to resolve an object reference for host:port {1} and has been marked unusable for future requests to that cluster.

Explanation: A Location Service Daemon has been marked unusable. In this case, a nodeagent is either unusable or is unreachable.

User Response: Check the status of the nodeagent. If the nodeagent had stopped for some reason, restart and continue. Otherwise, if the nodeagent is still running, check that it is reachable across the network, and then check its logs for possible errors.

- WWLM0064W: Errors have been encountered attempting to send a request to all members in cluster {0} and all of the members have been marked unusable for future requests to that cluster.

Explanation: All servers in a cluster either have been marked unusable or are not reachable.

User Response: Check the status of the cluster servers (stopped/start/running, etc.); if the cluster has stopped for some reason, restart and continue. Otherwise, if the servers are still running, check that they are reachable across the network, and then check their SystemOut/Err for possible errors.

- WWM0065W: An error was encountered attempting to update a cluster member {0} in cluster {1}, since it was not reachable from the Deployment Manager.

Explanation: In an attempt to refresh cluster information to servers thought to be active within a cluster, we found that a server is unreachable from the Deployment Manager.

User Response: Check the server status (stopped/start/running, etc.). If the server is down, restart and continue. If the server is still running, check that it is reachable across the network, and then check the server's SystemOut/Err for possible errors.

If any of these warning are encountered, follow the user response. If the warnings persist after following the user response, look at any other errors and warnings in the Log Analyzer on the affected servers.

You may also see exceptions with CORBA as part of the exception name, since WLM uses CORBA (Common Object Request Broker Architecture) to communicate between processes. Look for a statement in the exception stack specifying a "minor code." These codes denote the specific reason a CORBA call or response could not complete. WLM minor codes fall in the range of 0x4921040 - 0x492104F. For an explanation of minor codes related to WLM, see the Javadoc for the package and class `com.ibm.websphere.wlm.WsCorbaMinorCodes` in the InfoCenter:

<http://publib.boulder.ibm.com/infocenter/wasinfo/index.jsp>

2. You may analyze the PMI data to understand the workload arriving for each member of a cluster. The data for any one member of the cluster is only useful within the context of the data of all the members of the cluster.

You may get the PMI data using Tivoli Performance Viewer. The following counters are provided according to the Monitoring Level setting.

Table 9-1 Workload Management data counters

Name	Description	Granularity	Type	Level
numIncomingRequests	Total number of incoming IOP requests to an application server	per server	CountStatistic	Low
numIncomingStrongAffinityRequests	Number of incoming IOP requests to an application server that are based on a strong affinity. A strong affinity request is defined as a request that must be serviced by this application server because of a dependency that resides on the server. This request could not successfully be serviced on another member in the server cluster. In Version 5.0 ND edition, transactional affinity is the only example of a strong affinity.	per server	CountStatistic	Low
numIncomingNonAffinityRequests	Number of incoming IOP requests to an application server based on no affinity. This request was sent to this server based on workload management selection policies that were decided in the Workload Management (WLM) runtime of the client.	per server	CountStatistic	Low
numIncomingNonWLMObjectRequests	Number of incoming IOP requests to an application server that came from a client that does not have the WLM runtime present or where the object reference on the client was flagged not to participate in workload management.	per server	CountStatistic	Low
numServerClusterUpdates	Number of times initial or updated server cluster data is sent to a server member from the deployment manager. This metric determines how often cluster information is being propagated.	per server	CountStatistic	Low
numOfWLMClientServiced	Number of WLM enabled clients that have been serviced by this application server.	per server	CountStatistic	Low

Name	Description	Granularity	Type	Level
numOfConcurrentRequests	Number of remote IIOp requests currently being processed by this server.	per server	RangeStatistic	High
serverResponseTime	The response time (in milliseconds) of IIOp requests being serviced by an application server. The response time is calculated based on the time the request is received to the time when the reply is sent back out.	per server	TimeStatistic	Medium
numOfOutgoingRequests	The total number of outgoing IIOp requests being sent from a client to an application server.	per WLM	CountStatistic	Low
numClientClusterUpdates	The number of times initial or updated server cluster data is sent to a WLM enabled client from server cluster member. Use this metric to determine how often cluster information is being propagated.	per WLM	CountStatistic	Low
clientResponseTime	The response time (in milliseconds) of IIOp requests being sent from a client. The response time is calculated based on the time the request is sent from the client to the time the reply is received from the server.	per WLM	TimeStatistic	Medium

3. The JVM log is also a useful source to determine the problems.
4. You may collect a trace of both server and client.

For the server trace, please enable trace for all deployment manager, nodeagents and application servers with the Trace Specification as `WLM=all=enabled:ORBRas=all=enabled`.

For client trace, run `launchClient` with the `-Cctrace=true` option.

As mentioned above, the Workload Management plug-in to the Object Request Broker (ORB) distributes EJB requests, so the ORB trace is a very important resource when determining the workload management problem. For more information about ORB trace, please refer to 4.3, "WebSphere Application Server" on page 52.

9.1.5 HTTP session management

Session problems in cell environment will be complex. Before we can determine the problems, we need to briefly review some items.

1. Session tracking mechanism

There are several options for session tracking, depending on what sort of tracking method you want to use:

- Session tracking with cookies

This is the default. No special programming is required to track sessions with cookies.

- Session tracking with URL rewriting

An application that uses URL rewriting to track sessions must adhere to certain programming guidelines. The application developer needs to do the following:

- Program servlets to encode URLs
- Supply a servlet or Java Server Page (JSP) file as an entry point to the application

For more information about programming guidelines of URL rewriting, please refer to the InfoCenter at:

<http://publib.boulder.ibm.com/infocenter/wasinfo/index.jsp>

- Session tracking with SSL information

No special programming is required to track sessions with Secure Sockets Layer (SSL) information.

Because the SSL session ID is negotiated between the Web browser and HTTP Server, this ID cannot survive an HTTP Server failure. However, the failure of an application server does not affect the SSL session ID if an external HTTP Server is present between WebSphere Application Server and the browser. SSL tracking is supported for the IBM HTTP Server and iPlanet Web servers only. The internal HTTP Server of WebSphere also supports SSL tracking.

On the other hand, use either cookies or URL rewriting to maintain session affinity when using the SSL session ID as the session tracking mechanism in a cell environment.

2. Session management setting inheritance

There are three levels at which to set the session management: the Web container level, enterprise application level and Web module level.

By default, Web modules inherit Session Management settings from the application level above it, and applications inherit Session Management

settings from the Web container level above it. Except for the Web container level setting, there is an *Overwrite Session Management* checkbox, which specifies whether or not these Session Management settings take precedence over those normally inherited from a higher level for the current application or Web module.

3. Session scope

According to Servlet 2.3 specification, “*HttpSession objects must be scoped at the application (or servlet context) level. The underlying mechanism, such as the cookie used to establish the session, can be the same for different contexts, but the object referenced, including the attributes in that object, must never be shared between contexts by the container.*” That is to say, the Session Management facility supports session scoping through the Web module. Only servlets in the same Web module can access the data associated with a particular session.

On the other hand, WebSphere Application Server V5 provides an option that you can use to extend the scope of the session to an enterprise application. Therefore, you can share session attributes across all the Web modules in an enterprise application. This option is provided as an IBM extension. To share session data across Web modules in an enterprise application, check the **Shared httpsession context** checkbox on the IBM Extensions tab of the enterprise application you want to share in the Application Assembly Tool, as shown in Figure 9-11.

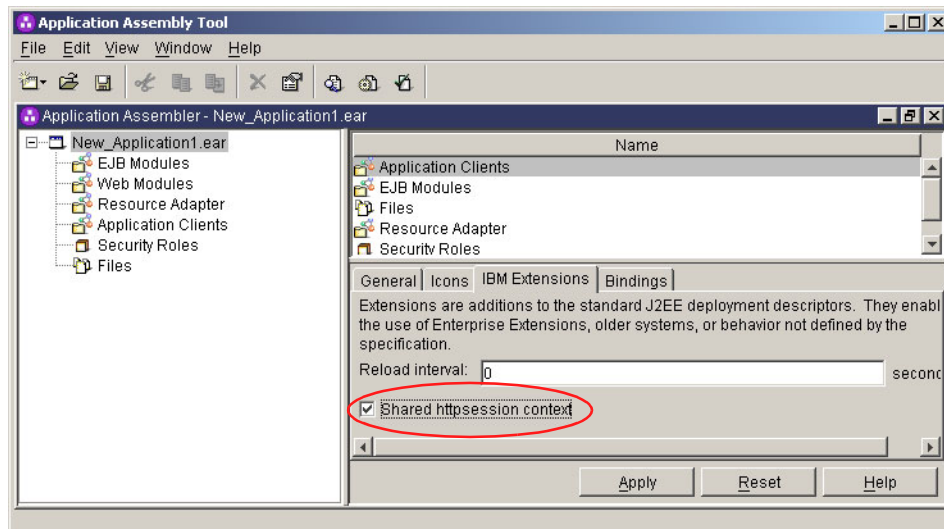


Figure 9-11 Sharing a session

There are some restrictions to use this option. You must install all the Web modules in the enterprise application on one given server, and you cannot split up Web modules in the enterprise application by servers. That is to say, you cannot install one Web module in one application server and another Web module in another application server if there is more than one Web modules in the enterprise application.

For enterprise applications on which this option is enabled, the Session Management configuration on the Web module inside the enterprise application is ignored. The Session Management configuration defined on the enterprise application is used if Session Management is overwritten at the enterprise application level. Otherwise, the Session Management configuration on the Web container is used.

4. Distributed sessions

For session recovery support, WebSphere Application Server provides distributed session support in a distributed environment. There are two distributed session mechanisms:

- Database Session persistence, where sessions are stored in the database specified.
- Memory-to-memory Session replication, where sessions are stored in one or more specified WebSphere Application Server instances.

Note: All the attributes set in a session must implement `java.io.Serializable` in the distributed session environment.

In general, consider making all objects held by a session serialized, even if immediate plans do not call for session recovery support.

Things to check

When you encounter a session problem, please consider the following items first.

1. Is the session scope that the customer wants in accordance with the application structure?

For example, the customer wants to share sessions across the Web modules, but the Shared `httpSession` context setting is not enabled when assembling the enterprise application, so the application cannot meet the customer's requirement.

2. Which level of Session Management setting is effective, Web container level, enterprise application level or Web module level? This is the basic question to answer before we determine whether the configuration is wrong.

3. Which session tracking mechanism is in use?

When using URL rewriting to track sessions, does the application adhere to the programming guidelines?

When SSL tracking is in use, is the Web server supported by WebSphere Application Server?

When tracking with cookies is in use, can the client accept cookies?

4. Do you understand the topology of the Distributed sessions setting?

Which kind of distributed sessions is in use, Database Session persistence or Memory-to-memory Session replication?

Which kind of runtime mode of Session Management is in use if Memory-to-memory Session replication is in use? Both Client and Server, Client only or Server only?

Symptom: Sessions are lost or not created

If you encounter a problem where sessions are lost or are not created, you may determine the exact nature of the problem by considering the following points.

1. Are there multiple Web modules within the enterprise application that encounter problems when tracking sessions?
 - If not, go to step 2 .
 - If so, do you want to share the session context across Web modules?
 - If so, do you set the Shared HttpSession context when assembling the application? If not, enable it.

Note: If the Shared HttpSession context option is enabled, the Session Management configuration on the Web module inside the enterprise application is ignored.

- Otherwise, make sure the Shared HttpSession context is not enabled. Do you then want to have different session settings among Web modules in an enterprise application?
 - If so, make sure the **Overwrite Session Management** checkbox of Web module level is checked in the Administrative Console.
 - Otherwise, make sure the Overwrite Session Management checkbox is not checked.
2. Check which level of Session Management setting is effective currently: Web container level, enterprise application level or Web module level?

Related settings include the *Overwrite Session Management* option of the enterprise application level and Web module level, which is set in the

WebSphere Application Server configuration, and the *Shared httpsession context* setting of enterprise application, which is set in the `ibm-application-ext.xmi`.

3. Which kind of client does the customer use? Does the client accept cookies?
 - If the client cannot accept cookies, did you enable the URL Rewriting mechanism in Session Management and does the application adhere to the programming guidelines?
 - Otherwise, please check the **Enable Cookies** setting in Session Management if the client can accept cookies.

Make sure the **Enable cookies** checkbox is checked in the Session Management page.

Verify the Cookies setting by clicking the **Enable cookies** link.

- Check the Cookie domain property.

This property controls whether or not a browser sends a cookie to particular servers. For example, if you specify a particular domain, session cookies are sent to hosts in that domain. The default domain is the server.

If the domain property is set, make sure it begins with a dot (.). Certain versions of Netscape do not accept cookies if the domain name does not start with a dot. Internet Explorer accepts the domain with or without a dot. For example, if the domain name is set to `mycom.com`, change it to `.mycom.com`.

On the other hand, if the cookie domain is set to `".mycom.com"`, resources should be accessed using that domain name, for example, `http://www.mycom.com/myapp/sessionServlet`. If you access the resources as `http://<ip_address>/myapp/sessionServlet`, a new session object will be created, but you cannot get the session in the following request.

- Check the Cookie path property.

This property specifies that a cookie is sent to the URL designated in the path. Specify any string representing a path on the server. `"/` indicates a root directory. Specify a value to restrict the paths to which the cookie will be sent. If you specify the root directory, the cookie is sent no matter which path on the given server is accessed.

Check whether the problematic URL is hierarchically below the Cookie path specified. If not, correct the Cookie path.

For example, you may have one Web module whose context root is `/test` and where the Cookie path is `/test/dir1`. A new session will be created if you access `/test/dir1/resours1`; the session attributes can

then be accessed when you access /test/dir1/dir2/resource2, but cannot be accessed when you access /test/resource.

- Check the Cookie maximum age property.

This property specifies the amount of time that the cookie lives on the client browser. Specify that the cookie lives only as long as the current browser session, or to a maximum age. If you choose the maximum age option, specify the age in seconds. The default is the current browser session which is equivalent to setting the value to -1.

If you set the maximum age, ensure that the client (browser) machine's date and time is the same as the server's, including the time zone. If the client and the server time difference is over the *Cookie maximum age* then every access would be a new session, since the cookie will expire after the access.

If you want to have different session settings among Web modules in an enterprise application which does not share the session context, ensure that each Web module specifies a different cookie name or path. If Web modules within an enterprise application use a common cookie name and path, ensure that the HTTP session settings, such as the Cookie maximum age, are the same for all Web modules. Otherwise, cookie behavior will be unpredictable and will depend upon which application creates the session. Note that this does not affect session data, which is maintained separately by the Web module.

4. Are you in a clustered environment, which means multiple nodes or multiple servers? If so, make sure that you have session persistence enabled.

You may continue with “Symptom: HTTP sessions are not persistent” on page 241.

Symptom: HTTP sessions are not persistent

If your HTTP sessions are not persistent, that is, session data is lost when the application server restarts or is not shared across the cluster, you may determine the problem according to the following steps.

1. Make sure which level of Session Management setting is effective to the problem Web module or enterprise application currently: Web container level, enterprise application level or Web module level?
2. Check to see which kind of distributed sessions you use: Database-based persistence or Memory-to-memory replication?

If Database-based persistence is in use

1. Check the Database Settings of Distributed Environment Settings by clicking the **Database** link.

- a. Check that the Datasource JNDI name is specified correctly.
- b. Check that the User ID and Password are specified correctly.
- c. If DB2 is in use, check that the DB2 row size matches the DB2 page size.
- d. If DB2 is in use, check that the Table space name is specified correctly when the DB2 Page Size is other than 4K.

Note: These settings have to be checked against the properties of an existing Data Source in the Administrative Console. The Session Manager does not automatically create a session database.

2. Check the related JDBC Providers setting and Data Sources setting.
 - a. Verify the scope of the JDBC Provider in use: Cell, Node, or Server?

Resources such as JDBC Providers can be defined at multiple scopes, with resources defined at more specific scopes overriding duplicates which are defined at more general scopes. The scope of the Data Source is the same as that of its JDBC Provider.

 - *Cell* is the most general scope. Resources defined at the Cell scope are visible from all nodes and servers, unless they are overridden.
 - Resources defined at the *Node* scope override any duplicates defined at the Cell scope and are visible to all servers on the same node, unless they are overridden at a Server scope on that node.
 - *Server* is the most specific scope for defining resources. Resources defined at the Server scope override any duplicate resource definitions defined at the Cell scope or parent Node scope and are visible only to a specific server.

When resources are created, the scope cannot be changed.

For example, the servers on other nodes cannot look up in the datasource which scope is on the specific node. If you want servers on other nodes to be able to look up a datasource with the same JNDI name, you have to create a JDBC Provider and a datasource in the cell scope, or create a JDBC Provider and datasource on each node.
 - b. Check the kind of JDBC Provider you have; it should not be XA enabled. For example, it should not be DB2 Legacy CLI-based Type 2 JDBC Driver (XA).
 - c. Check the Classpath and Native Library Path of the JDBC Provider. Check to see whether it is an absolute path or a Symbolic Link path with WebSphere variables, for example, `${DB2_JDBC_DRIVER_PATH}/db2java.zip`.

- If it is an absolute path:

If the JDBC Provider is in the cell scope, problems may occur if the operating systems of the related nodes are not the same or the JDBC driver classes of related nodes are not in same directory, because some servers may not be able to find the classes on the node according to the absolute path. A Symbolic Link path is recommended.

If the JDBC Provider is in Node scope or Server scope, make sure the JDBC Providers and datasource are created on each node or each server, and verify that the path locates the JDBC driver classes correctly.
 - If it is a Symbolic Link path:

First, verify which scope the variable has is effective: Cell, Node or Server.

If the variable in Cell scope is effective, problems may occur when the the operating systems of the related nodes are not the same or the JDBC driver classes of related nodes are not in the same directory. In order for each server to be able to find the classes in a particular classpath, define the variable in Node scope or Server scope.

Then, verify that the Symbolic Link path locates the JDBC driver classes correctly according to the value of variables.
- d. Check the JNDI Name of the datasource to see if it is the same as the setting of Database-based persistence.
 - e. Use the Test Connection function to make sure the datasource can be connected successfully in the Administrative Console.

If Memory-to-memory replication is in use

Verify that you have created the Internal Replication Domain and the Replicator Entry for each server. Verify that there is collision between the Replicator port and Client port of replicator entries on the same node.

Get more information

We cannot list all the problems of session management here, nor can we list all of the potential sources for each problem, but we can provide you with sources for more information if you do encounter a problem.

1. View the JVM logs (SystemOut.log and SystemErr.log) for the application server which hosts the problem application.

First, look at messages written while each application was starting. They will be written between the Starting application: <application> and Application started: <application> messages, as shown next.

Example 9-5 A sample of logs during application starting

```
[12/2/03 8:40:05:549 EST] 4276bb44 ApplicationMg A WSVR0200I: Starting
application: DefaultApplication.ear
[12/2/03 8:40:05:779 EST] 4276bb44 EJBContainerI I WSVR0207I: Preparing to
start EJB jar: Increment.jar
[12/2/03 8:40:06:210 EST] 4276bb44 EJBContainerI I WSVR0037I: Starting EJB jar:
Increment.jar
[12/2/03 8:40:06:951 EST] 4276bb44 ConnectionFac I J2CA0107I: Component-managed
authentication alias not specified for connection factory or datasource
Default_CF.
[12/2/03 8:40:07:391 EST] 4276bb44 WebContainer A SRVE0161I: IBM WebSphere
Application Server - Web Container. Copyright IBM Corp. 1998-2002
[12/2/03 8:40:07:441 EST] 4276bb44 WebContainer A SRVE0162I: Servlet
Specification Level: 2.3
[12/2/03 8:40:07:441 EST] 4276bb44 WebContainer A SRVE0163I: Supported JSP
Specification Level: 1.2
[12/2/03 8:40:07:782 EST] 4276bb44 WebContainer A SRVE0169I: Loading Web
Module: Default Web Application.
[12/2/03 8:40:08:142 EST] 4276bb44 WebGroup I SRVE0180I: [Default Web
Application] [/] [Servlet.LOG]: JSP 1.2 Processor: init
[12/2/03 8:40:08:563 EST] 4276bb44 WebGroup I SRVE0180I: [Default Web
Application] [/] [Servlet.LOG]: InvokerServlet: init
[12/2/03 8:40:08:593 EST] 4276bb44 WebGroup I SRVE0180I: [Default Web
Application] [/] [Servlet.LOG]: Snoop Servlet: init
[12/2/03 8:40:08:633 EST] 4276bb44 WebGroup I SRVE0180I: [Default Web
Application] [/] [Servlet.LOG]: Hello Pervasive Servlet: [Hello Pervasive
Servlet]init
[12/2/03 8:40:08:663 EST] 4276bb44 WebGroup I SRVE0180I: [Default Web
Application] [/] [Servlet.LOG]: Hit Count Servlet: init
[12/2/03 8:40:08:873 EST] 4276bb44 ApplicationMg A WSVR0221I: Application
started: DefaultApplication.ear
```

Within this block, look for any error or exception containing a package named `com.ibm.ws.webcontainer.httpsession`. If none is found, this is an indication that the Session Manager started successfully.

Look within the logs for any Session Manager related messages. These messages will be in the format `SESNxxxxE` and `SESNxxxxW` for errors and warnings, respectively, where `xxxx` is a number identifying the precise error. Look up the extended error definitions in the Session Manager message table in the WebSphere InfoCenter:

<http://publib.boulder.ibm.com/infocenter/wasinfo/index.jsp>

2. View the Process Logs (`native_stdout.log` and `native_stderr.log`) for the application server hosting the problem application.

WebSphere Application Server processes contain two output streams that are accessible to native code running in the process. These streams are the

stdout and stderr streams. By default, the stdout and stderr streams are redirected to log files at application server startup, which contain text written to the stdout and stderr streams by native modules.

By viewing these logs, we can find out whether there are problems when accessing native code.

Example 9-6 A sample of native_stderr.log

```
aCC runtime: Error 215 from
shl_findsym(/opt/WebSphere/AppServer/bin/libNBIO.sl,_shlInit)
/usr/lib/dld.sl: Unresolved symbol: typeid__XTQ2_3std9exception_(data) from
/opt/WebSphere/AppServer/bin/libNBIO.sl
/usr/lib/dld.sl: Unresolved symbol: __dt__Q2_3std9exceptionFv (code) from
/opt/WebSphere/AppServer/bin/libNBIO.sl
```

This is a sample showing that WebSphere Application Server 5.0.2 cluster members failed to start when replication entries were created in servers on HP-UX 11.11 or 11.00, which is fixed by PQ78548.

3. Monitor session status with Tivoli Performance Viewer.

To dynamically view the status of sessions as a Web application is running, enable the Performance Monitoring Service of the application server, then monitor the session status with Tivoli Performance Viewer.

The following counters can be used according to the Monitoring Level setting of Servlet Session Manager.

Table 9-2 Session Data Counters

Name	Description	Granularity	Type	Level
createdSessions	Number of sessions created	per Web application	CountStatistic	Low
invalidatedSessions	Number of sessions invalidated	per Web application	CountStatistic	Low
sessionLifeTime	The average session lifetime	per Web application	TimeStatistic	Medium
activeSessions	The number of concurrently active sessions. A session is active if WebSphere is currently processing a request which uses that session.	per Web application	RangeStatistic	High

Name	Description	Granularity	Type	Level
liveSession	The number of sessions that are currently cached in memory.	per Web application	RangeStatistic	High
NoRoomForNewSession	Applies only to session in memory with <code>AllowOverflow=false</code> . The number of times that a request for a new session cannot be handled because it would exceed the maximum session count.	per Web application	CountStatistic	Low
cacheDiscards	Number of session objects that have been forced out of the cache (an LRU algorithm removes old entries to make room for new sessions and cache misses). Applicable only for persistent sessions.	per Web application	CountStatistic	Low
externalReadTime	Time (milliseconds) taken in reading the session data from persistent store. For multirow sessions, the metrics are for the attribute; for single row sessions, the metrics are for the whole session. Applicable only for persistent sessions. When using a JMS persistent store, the user has the option to serialize the data being replicated. If they choose not to serialize the data, the counter will not be available.	per Web application	TimeStatistic	Medium

Name	Description	Granularity	Type	Level
externalReadSize	Size of session data read from persistent store. Applicable only for (serialized) persistent sessions; similar to externalReadTime above.	per Web application	TimeStatistic	Medium
externalWriteTime	Time (milliseconds) taken to write the session data from the persistent store. Applicable only for (serialized) persistent sessions. Similar to externalReadTime above.	per Web application	TimeStatistic	Medium
externalWriteSize	Size of session data written to persistent store. Applicable only for (serialized) persistent sessions. Similar to externalReadTime above.	per Web application	TimeStatistic	Medium
affinityBreaks	The number of requests received for sessions that were last accessed from another Web application. This can indicate failover processing or a corrupt plug-in configuration.	per Web application	CountStatistic	Low
serializableSessObjSize	The size in bytes of (the serializable attributes of) in-memory sessions. Only count session objects that contain at least one serializable attribute object. Note that a session may contain some attributes that are serializable and some that are not. The size in bytes is at a session level.	per Web application	TimeStatistic	Max

Name	Description	Granularity	Type	Level
timeSinceLastActivated	The time difference in milliseconds between previous and current access time stamps. Does not include session time out.	per Web application	TimeStatistic	Medium
invalidatedViaTimeout	The number of requests for a session where no CountStatistic exists, presumably because the session timed out.	per Web application	CountStatistic	Low
attemptToActivateNotExistentSession	Number of requests for a session that no longer exists, presumably because the session timed out. Use this counter to help determine if the timeout is too short.	per Web application	CountStatistic	Low

You may set different Monitoring Levels according to your requirements, and this will give you an indication of the nature of the problem, for example, whether or not sessions are actually being created.

For more information about the usage of Tivoli Performance Viewer, please refer to 4.3, “WebSphere Application Server” on page 52 .

- Of course, you can get session management configuration information in the Administrative Console and monitor the session status in Tivoli Performance Viewer, but there is an alternative way to display the current configuration and statistics related to session tracking, which is a special servlet. This servlet has all the counters that are in the performance monitor tool and has some additional counters as well.

The servlet name is `com.ibm.ws.webcontainer.httpsession.IBMTrackerDebug`. It can be invoked from any Web module which is enabled to serve by class name, for example, using `default_app`:

```
http://localhost:9080/servlet/com.ibm.ws.webcontainer.httpsession
.IBMTrackerDebug
```

If you are viewing the module via the Serve by class name feature, be aware that it may be viewable by anyone who can view the application. You may wish to map a specific, secured URL to the servlet instead and disable the Serve by class name feature.

5. Enable tracing for the HTTP Session Manager component.

For information about how to enable trace, please refer to 4.3, “WebSphere Application Server” on page 52.

Use the trace specification

`com.ibm.ws.webcontainer.httpsession.*=all=enabled` to trace the Session Manager activity. If you are using persistent sessions based on memory replication, enable trace for `com.ibm.ws.drs.*` also.

6. Inspect the cookies in browser using `javascript:alert(document.cookie)`.

Input `javascript:alert(document.cookie)` into the Address field of the browser and then press **Enter**; a message window will appear and the current cookies will be listed, as shown in Figure 9-12 .



Figure 9-12 Cookie information

When you access more information, you can analyze the problem, search for it in the WebSphere support page

(<http://www-306.ibm.com/software/webservers/appserv/was/support/>), or provide the information to the IBM support team to receive further help.

9.1.6 Resources

There are many resources in WebSphere Application Server, and most problems related to resources are discussed in 7.1.8, “Resource providers” on page 152 and Chapter 8, “WebSphere Application Server Enterprise” on page 173. We would, however, like to remind you to pay attention to the scope of the resources and WebSphere variables in this chapter.

We use the JDBC Provider here as an example.

When you are working in the Network Deployment Administrative Console in a cell, it is important to apply the correct scope to the operation that you wish to perform. Otherwise, you may define an environment setting or resource incorrectly. For example, if you set the value of the `DB2_JDBC_DRIVER_PATH` variable using the Cell scope, all nodes in the cell will look for the driver in the same location. If not all nodes are installed within the same directory structure,

this will not work. In other words, in a cell environment, all the nodes have to have the same exact directory structure as the cell for all the subsequent drivers and links.

If the nodes are a mixture of platforms, then follow the hints provided below.

When setting the scope at the Cell level, you should be using a Symbolic Link and not an absolute path. If the path specified for the datasource driver is specified as an absolute path and it is not found on one node, then the startup will fail. But, if you use the Symbolic Link (DB2_JDBC_DRIVER_PATH), which points to the driver name and not an absolute path (for example: DB2_JDBC_DRIVER_PATH/db2java.zip), then that Symbolic Link value can be set differently on each node.

If the Deployment Manager test connection does not work, it does not necessarily indicate a problem with the connection between the database and the application servers. The Deployment Manager tests the connection from the machine where it is installed. If you do not have the Deployment Manager machine configured to access the database, the test connection will fail, but the application servers may use the database connection with no problem. Applications will pick up the drivers from their respective nodes.

9.1.7 JMS Provider

If you installed the embedded messaging server feature on the base node, when you add it to a cell, the the Deployment Manager instantiates the JMS provider provided by WebSphere Application Server; this is the jmsserver process.

Things to check

1. Review the JVM log for the jmsserver.
2. Check for port conflicts if the jmsserver is not started. If you have a coexistence configuration, you have to manually change the default port settings after federating a node.
3. If you are running the nodeagent as a non-root user, the jmsserver needs to run under the non-root user ID.

Symptom: InvalidExecutableException while starting jmsserver

If you installed Network Deployment first and then WebSphere Application Server with embedded messaging on the same node, you will see this error. The correct sequence of events is to install WebSphere Application Server first and then Network deployment.

Symptom: jmsserver fails to start after configuring as non-root

If you configure the jmsserver on AIX or Solaris to use a non-root user ID, the jmsserver may not start due to invalid semaphores and shared memory segments that were created for the root user. For more information on recovering from this, see the Technote *The WebSphere Application Server 5.0 Embedded JMS Broker cannot be started when Embedded JMS is configure to run with Non-root User ID*, at:

http://www-1.ibm.com/support/docview.wss?rs=180&context=SSEQTP&q1=jmsserver&uid=swg21144021&loc=en_US&cs=utf-8&lang=en

Check that the jmsserver has been started on the node specified in the QueueConnectionFactory or TopicConnectionFactory. The error message in the SystemOut.log file will look like this:

```
javax.jms.InvalidDestinationException: MQJMS2008: failed to open MQ queue
linked exception com.ibm.mq.MQException: MQJE001: Completion Code 2,
Reason 2085
```

9.1.8 Naming

Naming is used by clients of WebSphere Application Server applications most commonly to obtain references to objects related to those applications, such as Enterprise JavaBean (EJB) homes, or to obtain references to objects related to named resources, such as datasources.

There are some new features in the WebSphere Application Server V5 naming implementation. For additional scalability, the name space for a cell is distributed among various servers. The name space for the entire cell is federated among all servers in the cell. Every server process contains a name server. All name servers provide the same logical view of the cell name space. The various server roots and persistent partitions of the name space are interconnected by means of a system name space. This allows you to get to any context in the cell name space. In previous releases, there was only one name server for an entire administrative domain.

The name space may cause problems if you need to work across servers or clusters.

There are two terms that should be clarified first.

- ▶ Relative names

All names are relative to a context. A name which can be resolved from one context in the name space, but cannot necessarily be resolved from another context in the name space is a relative name.

► Qualified names

All names are relative to a context. Here, the term *qualified name* refers to names that can be resolved from any initial context in a cell. This action is accomplished by using names that navigate to the same context, the cell root. All qualified names begin with the string `cell/` to navigate from the current initial context back to the cell root context.

For example, an object bound in a single server has a topology-based qualified name of the following form:

```
cell/nodes/<node_name>/servers/<server_name>/<relative_Jndi_name>
```

An object bound in a server cluster has a topology-based qualified name of the following form:

```
cell/clusters/<cluster_name>/<relative_Jndi_name>
```

If you are accessing a JNDI name from one server to another server, you may experience naming problems if you do not have a fully qualified name.

Also review 7.1.17, “Naming” on page 168.

Things to check

1. Make sure that the target server can be accessed from the JNDI client.
 - a. From a command prompt on the client's server, enter `ping servername` and verify the connectivity.
 - b. Use the WebSphere Application Server Administrative Console to verify that the target resource's application server and, if applicable, EJB module or Web module, are started.
2. Which kind of JNDI client is this?
 - a. If the JNDI client is a JSP, servlet or EJB:

Does it run in the same process as the named object?

 - i. If the JNDI client runs in the same process as the named object, does it look up the object in the local namespace or the global namespace? In other words, does it look up the object as `java:comp/env` or not?

If the local namespace is used, that is to say, the object is looked up as `java:comp/env`, which name is the reference bound to in the deployment descriptor? Does the bound name exist?

If the global namespace is used, does the name exist?
 - ii. If the JNDI client does not run in the same process as the named object, does it look up the object in the local namespace or the global namespace? In other words, does it look up the object as `java:comp/env` or not?

If the local namespace is used, that is to say, the object is looked up as `java:comp/env`, which name is the reference bound to in the deployment descriptor? Is the bound name a qualified name? If it is not a qualified name, correct it.

If the global namespace is used, is it a qualified name? If it is not a qualified name, correct it.

- b. If the JNDI client is a J2EE client:

Does the JNDI client look up an object that resides on a server different from the one from which the initial context was obtained?

- i. If so, does it look up the object in the local namespace or the global namespace? In other words, does it look up the object as `java:comp/env` or not?

If the local namespace is used, that is to say, the object is looked up as `java:comp/env`, which name is the reference bound to in the deployment descriptor? Does the bound name exist?

If the global namespace is used, does the name exist?

- ii. If not, does it look up the object in the local namespace or the global namespace? In other words, does it look up the object as `java:comp/env` or not?

If the local namespace is used, that is to say, the object is looked up as `java:comp/env`, which name is the reference bound to in the deployment descriptor? Is the bound name a qualified name? If it is not a qualified name, correct it.

If the global namespace is used, is it a qualified name? If it is not a qualified name, correct it.

- c. If the JNDI client is a thin client, which name does the client look up? The name must be a qualified name.

3. Make sure the name which the JNDI client looks up exists.

Symptom: CannotInstantiateObjectException from JNDI lookup operation

Follow the steps in “Things to check” on page 252. Verify that you are using a correct name. In particular, a fully qualified name must be used if the JNDI client looks up an object that resides on a server different from the one from which the initial context was obtained. Check that the server that hosts the JNDI name is running. Use `dumpNameSpace` to verify that the JNDI name exists.

Symptom: NameNotFoundException from JNDI lookup operation

If you encounter this exception while trying to access an enterprise bean, data source, messaging resource, or other resource, possible causes include:

- ▶ A serialized Java object is being looked up, but the necessary classes required to deserialize it are not in the runtime environment.
- ▶ A Reference object is being looked up, and the associated factory used to process it as part of the lookup process is failing.

To determine the precise cause of the problem:

- ▶ Look in the JVM logs of the server hosting the target resource. Look for exceptions immediately preceding the `CannotInstantiateObjectException`. If it is a `java.lang.NoClassDefFoundError` or `java.lang.ClassNotFoundException`, make sure the class referenced in the error message can be located by the class loader.
- ▶ Print out the stack trace for the root cause and look for the factory class. It will be called `javax.naming.NamingManager.getObjectInstance()`. The reason for the failure will depend on the factory implementation, and may require you to contact the developer of the factory class.

Symptom: OperationNotSupportedException from JNDI Context operation

This error has two possible causes:

- ▶ An update operation, such as a bind, is being performed with a name that starts with `java:comp/env`. This context and its subcontexts are read-only contexts.
- ▶ A context bind or rebind operation of a non-CORBA object is being performed on a remote name space that does not belong to WebSphere Application Server. Only CORBA objects can be bound to these CosNaming name spaces.

To determine which of these errors is causing the problem, check the full exception message.

Get more information

1. Review the JVM logs to see what name is not found and what code is looking for the name.
2. Use the `dumpNameSpace` tool to view JNDI names.

3. To run trace, use `com.ibm.ws.naming.* :com.ibm.websphere.naming.*` as the trace specification.

9.1.9 ORB

IBM ORB (RMI/IIOP) is an important part of WebSphere. Most of the communication between the components of WebSphere takes place through RMI-IIOP. Any error within the ORB can be fatal to the application running within WebSphere.

Identifying failure in ORB

A stack trace generated at the point of failure will give an indication of whether or not the cause of the failure is ORB. If the JVM log file contains exceptions that include words like "CORBA" and "rmi" many times, the problem could be related to ORB. These exceptions can be divided into two types:

1. *User exceptions* are IDL-defined and inherit from `org.omg.CORBA.UserException`. They are not usually fatal and should always be handled by the application. In most cases, ORB will not be the source of the problem
2. *System exceptions* are thrown by the application and show an unusual condition where ORB is not able to recover gracefully. All system exceptions belong to the `org.omg.CORBA` package. The most common system exceptions are:
 - BAD_OPERATION
 - BAD_PARAM
 - COMM_FAILURE
 - MARSHAL
 - UNKNOWN

Each system exception has two pieces of information, a completion status and a minor code. The completion status will say whether or not the attempted operation was completed successfully. The minor code is a long integer and can be used to store any vendor-specific value. Minor code can be very helpful in identifying the exact location of the ORB code where the exception is thrown when the stack trace is missing.

If you suspect that the source of the failure is ORB, you need to collect the ORB traces by recreating the problem.

Almost all of the ORB code is written in Java; therefore, the chances of getting a platform-dependent problem in ORB are very minimal. If you suspect that the cause of the problem is ORB, try the testcase on multiple platforms to confirm your assumption.

Trace facility in ORB

There are two parts to the trace facility in ORB.

1. The ORB trace is used for tracing the ORB builds. Every method in the ORB that does tracing declares the `String methodName`, since this will be used by the trace to show which method is being executed. Exceptions that are caught and thrown are always logged. This will preserve the record of the original exception thrown by the JVM. Exception chaining introduced in JDK 1.4 provides a facility to preserve the history of exceptions without every exception being logged. A typical enabled trace looks like this.

Example 9-7 ORB trace sample

```
11:47:34.901 com.ibm.rmi.util.Version logVersions:110 P=854740:0=0:CT ORBRas[default] IBM Java
ORB build orbn131-20030530
11:47:34.901 com.ibm.rmi.util.Version logVersions:117 P=854740:0=0:CT ORBRas[default] J2RE
1.3.1 IBM Windows 32 build cn131-20030430 (JIT enabled: jitc)
11:47:34.901 com.ibm.CORBA.iiop.ORB parseProperties:1802 P=854740:0=0:CT ORBRas[default] ORB
Native Codeset (char) = ISO8859_1 (10001)
11:47:34.901 com.ibm.CORBA.iiop.ORB parseProperties:1808 P=854740:0=0:CT ORBRas[default] ORB
Native Codeset (wchar) = UCS2 (10100)
11:47:34.901 com.ibm.CORBA.iiop.ORB parseProperties:2084 P=854740:0=0:CT ORBRas[default] The
value of the property com.ibm.CORBA.MaxOpenConnections" is null The highWaterMark has been set
to 240
11:47:34.901 com.ibm.CORBA.iiop.ORB parseProperties:2112 P=854740:0=0:CT ORBRas[default] The
value of the property "com.ibm.CORBA.MinOpenConnections" is null The lowWaterMark has been set
to 100
11:47:34.911 com.ibm.rmi.iiop.BufferPool <init> (int, int):84 P=854740:0=0:CT ORBRas[default]
DefaultBufferPool: Initialize BufferPool - BufferPool max size: 100 Buffer length: 1024
```

2. The `CommTrace` is used to trace the flow of information across the wire.

This can be done regardless of whether or not the main trace is on.

`CommTrace` is used to understand the `GIOP()` message flow across the wire between client and server.

Example 9-8 CommTrace sample

```
OUT GOING:
Request Message
Date:          November 7, 2003 11:47:35 AM IST
Thread Info:   P=854740:0=0:CT
Local Port:    1337 (0x539)
Local IP:      9.184.197.173
Remote Port:   2809 (0xAF9)
Remote IP:     9.184.197.173
GIOP Version:  1.2
Byte order:    big endian
Fragment to follow: No
```

```

Message size: 321 (0x141)
--
Request ID:      6
Response Flag:   WITH_TARGET
Target Address:  0
Object Key:     length = 22 (0x16)
                4C4D4249 00000015 B4A1420A 00100000
                00040000 0000
Operation:      resolve
Service Context: length = 4 (0x4)
  Context ID:   1229081874 (0x49424D12)
  Context data: length = 8 (0x8)
                00000000 13100005
  Context ID:   17 (0x11)
  Context data: length = 2 (0x2)
                0002
  Context ID:   1 (0x1)
  Context data: length = 12 (0xC)
                00000000 00010001 00010100
  Context ID:   6 (0x6)
  Context data: length = 178 (0xB2)
                00000000 00000028 49444C3A 6F6D672E
                6F72672F 53656E64 696E6743 6F6E7465
                78742F43 6F646542 6173653A 312E3000
                00000001 00000000 00000076 00010200
                0000000E 392E3138 342E3139 372E3137
                3300053A 0000001A 4C4D4249 00000010
                5189B962 00100000 00080000 00000000
                00000000 00000003 00000001 00000018
                00000000 00010001 00000001 00010020
                00010100 00000000 49424D0A 00000008
                00000000 13100005 00000026 00000002
                0002
Data Offset:    12e

0000: 47494F50 01020000 00000141 00000006  GIOP.....A....
0010: 03000000 00000000 00000016 4C4D4249  .....LMBI
0020: 00000015 B4A1420A 00100000 00040000  .....B.....
0030: 00000000 00000008 7265736F 6C766500  .....resolve.
0040: 00000004 49424D12 00000008 00000000  ....IBM.....
0050: 13100005 00000011 00000002 00020000  .....
0060: 00000001 0000000C 00000000 00010001  .....
0070: 00010100 00000006 000000B2 00000000  .....
0080: 00000028 49444C3A 6F6D672E 6F72672F  ... (IDL:omg.org/
0090: 53656E64 696E6743 6F6E7465 78742F43  SendingContext/C
00A0: 6F646542 6173653A 312E3000 00000001  odeBase:1.0....
00B0: 00000000 00000076 00010200 0000000E  .....v.....
00C0: 392E3138 342E3139 372E3137 3300053A  9.184.197.173..:
00D0: 0000001A 4C4D4249 00000010 5189B962  ....LMBI....Q..b

```

```

00E0: 00100000 00080000 00000000 00000000 .....
00F0: 00000003 00000001 00000018 00000000 .....
0100: 00010001 00000001 00010020 00010100 .....
0110: 00000000 49424D0A 00000008 00000000 ....IBM.....
0120: 13100005 00000026 00000002 00020000 .....&.....
0130: 00000001 0000000D 48656C6C 6F536572 .....HelloSer
0140: 76696365 00000000 00000001 00 vice.....

```

Elements of the GIOP Request Message are as follows:

1. **OUTGOING** - This means that this message is going out to the process which generated this trace file.
2. **Request Message** is a GIOP request message. There are eight types of GIOP messages. Two ORBs communicate with each other using these eight GIOP messages:
 - Request Message
 - Reply Message
 - Cancel Request Message
 - Locate Request Message
 - Locate Reply Message
 - Close Connection Message
 - Message Error
 - Fragment Message
3. **Local Port** - The port from which this message is being sent out.
4. **Local IP** - The IP address of the local machine.
5. **Remote Port** - The port to which this message is being sent.
6. **Remote IP** - The IP address of the remote machine. In the above example, the local process and the remote process are running in the same machine, hence the same IP address.
7. **GIOP Version** - The specification version of GIOP being supported by the local process. If either receiver is at a lower level of GIOP, further communication is carried out in the lower level of GIOP.
8. **Byte Order** - Determines whether byte ordering is Big Endian or Little Endian.
9. **Fragment to Follow** - Determines whether or not any fragmented message is following this message. The value No means that this is the last fragment of the message.
10. **Message Size** - The size of the message being sent out.
11. **Request ID** - The ID number of the message being sent out.
12. **Response Flag** - WITH_TARGET means that a reply is expected for this message.

13. Target Address - The target address.
14. Object key - The object key is created by the server.
15. Operation - The remote operation to be carried out at the remote location.
16. Service Context - Service Contexts are used to pass extra information on the flow. A Service Context consists of an ID, followed by an arbitrary sequence of bytes of data. IDs are 16-bit Hex, and are assigned by the OMG. IBM has its own set of IDs which all start with 0x49424D ("IBM" in Hex). Commonly used service contexts are codesets and IBM's ORB/Partner version (0x49424D12, sent on every request). Codeset service context is used to negotiate the codeset used to transmit data.
17. Service Contexts: There are four service contexts.
18. The first 12 bytes in the message (as shown below) are the GIOP header and the rest is the message body:

```
0000: 47494F50 01020000 00000141 00000006  GIOP.....A....
```

GIOP message is used by the ORB developer to figure out what went through the wire. This makes the debugging of ORB easier.

Enabling ORB tracing

Follow these steps to enable the tracing facility of ORB for standalone Java applications:

1. The ORB trace can be switched on by setting the following property:
-Dcom.ibm.CORBA.Debug=true
2. The CommTrace or the communication trace in ORB can be switched on by setting the following property:
-Dcom.ibm.CORBA.CommTrace=true
3. The trace output can be stored to a desired file by setting the following property:
-Dcom.ibm.Debug.Output=<filename>

Enabling the tracing facility of ORB for a server running in WebSphere can be done using either of the two methods given below:

1. In the server.xml located in
WebSphere\AppServer\config\cells\northray\nodes\<node name>\servers\<server_name> directory, set: commTraceEnabled="true".
2. From the Administrative Console, click **Servers -> Application Servers -> <server_name> -> Configuration -> (Additional Properties) ORB Service**, and select the box **ORBTracing**. This enables both the ORB Trace and

Comm trace. The output goes to the trace.log file in the <WebSphere_root>/logs/<server_name> directory.

Both of the above methods will turn on ORB Trace as well as CommTrace.

Symptom: Marshal Exception

If the stack trace or ORB trace has an omg.orb.CORBA.MARSHAL system exception, the string following the exception will contain details about the exception along with the completion status and minor code.

Symptom: Server Hangs

A server hang usually happens for one of the following reasons:

- ▶ A deadlock between two or more threads.
 - ▶ Waiting for an I/O operation.
 - ▶ 100% CPU usage.
1. The first thing to check when experiencing a hang is whether or not the application is running on multiple CPU machines. If the hang occurs in a multi-processor scenario and disappears if multiple CPUs are not used, you may have a synchronization problem.

ORB could be waiting for a read/write operation; this looks like a hang. The following ORB properties should be checked for this condition:

- com.ibm.CORBA.RequestTimeout=<NNNN> - If this value is set, ORB throws an exception once the time period to wait for a reply to a request message is elapsed. The exception will contain a string with an explanation.
 - com.ibm.CORBA.FragmentTimeout=<NNNN> - If this value is set, ORB will throw an exception once the time period to wait for the next fragment message is elapsed.
 - com.ibm.CORBA.LocateRequestTimeout=<NNNN> - If this value is set, ORB will throw an exception once the time period to wait for the Locate Reply Message for the LocateRequest Message sent by the server is elapsed.
2. Next, check whether the CPU usage is approaching 100%. This could be due to looping on a variable.
 3. Javacore should be collected along with ORB traces and any relevant information in each case specified above.

Symptom: Fragmentation problems

When the client/server is sending a very large amount of data across the wire, it is sent as fragment messages. Problems could occur when fragments are written to or read from a stream. To confirm this, fragmentation can be switched off by

setting the property `com.ibm.CORBA.FragmentSize=0`. If the problem persists, you can rule out a fragmentation issue.



WebSphere MQ

Distributed environments often use some sort of message-oriented middleware to provide messaging and asynchronous capabilities between application and system components.

In our eMerge scenario, WebSphere MQ provides the messaging middleware for the applications.

This chapter focuses on the common problems in a WebSphere MQ environment, specifically in a messaging scenario with application servers and message broker.

10.1 WebSphere MQ

WebSphere MQ, for most installations, is run as a client/server application which incorporates many different platforms and operating systems that interact with each other. You will often need to review data from many of the components that make up the WebSphere MQ environment. These could include other z/OS systems, UNIX systems from many other vendors, or Windows. The most important diagnostic data to review is written to the MQ error log and the associated FDC files. These diagnostic files may be created on both the server (where the Queue Manager resides) and on the client, so it is important to review both client and server for the relevant files.

Things to check

1. Has WebSphere MQ run successfully before?

If not, verify the installation and configuration of WebSphere MQ. Verify the post-installation configuration also.

2. Are there any error messages?

WebSphere MQ uses error log to capture messages concerning the operation of WebSphere MQ, any queue managers start, error data coming from the channels that are in use. Verify the error logs to see whether any messages associated with the above is present.

3. Is there any return code?

If your application gets a return code indicating that the Message Queue Interface (MQI) call has failed, refer to the *WebSphere MQ Application Programming Reference Manual* for the description of that return code.

4. Can the problem be reproduced?

If so, try to recreate it to determine whether it is an application problem or System problem.

5. Have any changes been made since the last successful run?

- Did you try to identify the changes made recently?
- Have queue or channel definitions been added, changed, or deleted recently?
- Has the application which handles the return code been changed recently?
- Have any of the operating systems that can affect the WebSphere MQ operation been changed, like the registry entry in Windows, etc.?

6. Has the application run successfully before?

If so, verify whether any change have been made in the application. Is it possible to retry using a back level of the application? Have all the features of the application been tested properly? Does the application run on other WebSphere MQ systems?

7. Problems with commands

You should be careful when including special characters, for example, backslash(\) and double quote (“) characters, in descriptive text for some commands. If you use either of these characters in descriptive text, precede them with a \ (backslash), that is, enter \\ (double backslash) or \" (backslash and double quote) if you want \ (backslash) or “ (double quote) in your text.

8. Does the problem affect specific parts of the network?

If the link to a remote message queue manager is not working, the messages cannot flow to the remote queue. Check that communication between the two systems is available, and that the intercommunication component of WebSphere MQ has been started. Check that messages are reaching the transmission queue, and check the local queue definition of the transmission queue and any remote queues. Have any network related changes been made? Have any WebSphere MQ definitions been changed?

9. Does the problem occur at specific times of the day?

This could be due to system loading. If your WebSphere MQ network extends across more than one time zone, you may reach a peak in system load.

10. Is the problem intermittent?

An intermittent problem could be caused by ignoring the fact that processes can run independently of each other (for example, issuing a **MQGET** call without specifying a `wait` option before completing an earlier process).

11. Have all the latest service updates been applied?

10.1.1 WebSphere MQ Messaging issues with the WebSphere Application Server

The following symptoms are related to WebSphere Application Server and WebSphere MQ integration.

Symptom: Failed to receive a response from a PCF command

If you have issued a command but you have not received a response, consider the following:

- ▶ Is the command server running?

Work with the **dspmqcsv** command to check the status of the command server. If the response to this command indicates that the command server is not running, use the **strmqcsv** command to start it. If the response to the command indicates that the SYSTEM.ADMIN.COMMAND.QUEUE is not enabled for **MQGET** requests, enable the queue for **MQGET** requests.

- ▶ Has a reply been sent to the dead-letter queue?

The dead-letter queue header structure contains a reason or feedback code describing the problem. If the dead-letter queue contains messages, you can use the provided browse sample application (amqsbcbg) to browse the messages using the **MQGET** call.

- ▶ Has a message been sent to the error log?
- ▶ Are the queues enabled for put and get operations?
- ▶ Is the wait interval long enough?

If your **MQGET** call has timed out, a completion code of MQCC_FAILED and a reason code of MQRC_NO_MSG_AVAILABLE are returned. If you are using your own application program to put commands onto the SYSTEM.ADMIN.COMMAND.QUEUE, do you need to take a syncpoint? Unless you have specifically excluded your request message from syncpoint, you need to take a syncpoint before attempting to receive reply messages.

- ▶ Have MAXDEPTH and MAXMSGL attributes of your queues been set at a sufficiently high value?
- ▶ Are you using the CorrelId and MsgId fields correctly?

Set the values of MsgId and CorrelId in your application to ensure that you receive all messages from the queue.

Try stopping the command server and then restarting it, responding to any error messages that are generated. If the system still does not respond, the problem could be with either a queue manager or the whole of the WebSphere MQ system. First, try stopping individual queue managers to attempt to isolate a failing queue manager. If this does not reveal the problem, try stopping and restarting WebSphere MQ, responding to any messages that are generated in the error log.

Symptom: Some of the queues are failing

If you suspect that the problem occurs with only a subset of queues, check the local queues that you think are having problems. Display the information about each queue. You can use the MQSC command **DISPLAY QUEUE** to display the information. Use the data displayed to run the following checks:

- ▶ If CURDEPTH is at MAXDEPTH, this indicates that the queue is not being processed. Check that all applications are running normally.
- ▶ If CURDEPTH is not at MAXDEPTH, check the following queue attributes (if triggering is being used).
 - Is the trigger depth too high?
 - Is the process name correct?
 - Is the process available and operational?
 - Can the queue be shared? If not, could another application already have it open for input?
 - Is the queue enabled appropriately for **GET** and **PUT**?
 - Is the trigger monitor running?
- ▶ Verify whether any application processes are getting messages from the queue, and if not, determine why this is so. It could be because the applications need to be started, a connection has been disrupted, or the **MQOPEN** call has failed for some reason. Check the queue attributes **IPPROCS** and **OPPROCS**. These attributes indicate whether the queue has been opened for input and output. If the value is zero, it indicates that no operations of that type can occur.

Note: Note that the values may have changed and that the queue was open but is now closed. You need to check the status at the time you expect to put or get a message.

Symptom: Problem affects only remote queues

If the problem affects only remote queues, check the following:

- ▶ Check that required channels have been started, that they can be triggered, and that any required initiators are running.
- ▶ Check that the programs that should be putting messages to the remote queues have not reported problems.
- ▶ If you use triggering to start the distributed queuing process, check that the transmission queue has triggering set to on. Also, check that the trigger monitor is running.
- ▶ Check the error logs for messages indicating channel errors or problems. If necessary, start the channel manually.

Symptom: Application or system running slowly

If your application is running slowly, this could indicate that it is in a loop, or waiting for a resource that is not available.

This could also be caused by a performance problem. Perhaps it is because your system is operating near the limits of its capacity. This type of problem is probably worst at peak system load times, typically in mid-morning and mid-afternoon. A performance problem may be caused by a limitation of your hardware.

Note: If your network extends across more than one time zone, peak system load might seem to occur at some other time.

If you find that performance degradation is not dependent on system loading, but happens sometimes when the system is lightly loaded, a poorly designed application program is probably to blame. This could manifest itself as a problem that only occurs when certain queues are accessed.

Symptom: The obtained output is incorrect

In this document, incorrect output refers to your application:

- ▶ Not receiving a message that it was expecting.
- ▶ Receiving a message containing unexpected or corrupted information.
- ▶ Receiving a message that it was not expecting, for example, one that was destined for a different application.

In all cases, check that any queue or queue manager aliases that your applications are using are correctly specified and accommodate any changes that have been made to your network. If a WebSphere MQ error message is generated, look at all the log which are prefixed with the letters AMQ.

Additional problems that you might find if your application includes the use of distributed queues are discussed below.

- ▶ Has the message been put on the queue successfully?
 - Has the queue been defined correctly? For example, is MAXMSGL sufficiently large?
 - Is the queue enabled for putting?
 - Is the queue already full? This could mean that an application was unable to put the required message on the queue.
 - Has another application got exclusive access to the queue?
 - The message may be too long.
 - Perhaps the sender is not authorized to use the destination queue.
 - Perhaps even message puts have been inhibited on the destination queue.

- ▶ Are you able to get any messages from the queue?
 - Do you need to take a syncpoint? If messages are being put or retrieved within syncpoint, they are not available for other tasks until the unit of recovery has been committed.
 - Is your wait interval long enough? You can set the wait interval as an option for the **MQGET** call. You should ensure that you are waiting long enough for a response.
 - Are you waiting for a specific message that is identified by a message or a correlation identifier (MsgId or CorrelId)? Check that you are waiting for a message with the correct MsgId or CorrelId. A successful **MQGET** call sets both these values to that of the message retrieved, so you may need to reset these values in order to get another message successfully.
- ▶ Can you get other messages from the queue?
 - Can other applications get messages from the queue?
 - Was the message you were expecting defined as persistent? If not and WebSphere MQ has been restarted, the message has been lost.
 - Has another application got exclusive access to the queue?
- ▶ If the queue is functioning correctly:

If you are unable to find anything wrong with the queue, and WebSphere MQ is running, run the following checks on the process through which you expected to put the message on to the queue:

- Did the application get started? If it should have been triggered, check that the correct trigger options were specified.
- Did the application stop?
- Is a trigger monitor running?
- Was the trigger process defined correctly?
- Did the application complete correctly?
- Look for evidence of an abnormal end in the job log. Did the application commit its changes, or were they backed out?

If multiple transactions are serving the queue, they can conflict with one another. For example, suppose one transaction issues an **MQGET** call with a buffer length of zero to find out the length of the message, and then issues a specific **MQGET** call specifying the MsgId of that message. However, in the meantime, another transaction issues a successful **MQGET** call for that message, so the first application receives a reason code of **MQRC_NO_MSG_AVAILABLE**. Applications that are expected to run in a multiple server environment must be designed to cope with this situation.

- Consider that the message could have been received, but that your application failed to process it in some way. For example, did an error in the expected format of the message cause your program to reject it?
- ▶ Messages that contain unexpected or corrupted information:

If the information contained in the message is not what your application was expecting, or has been corrupted in some way, consider the following points:

 - Has your application, or the application that put the message onto the queue, changed? Ensure that all changes are simultaneously reflected on all systems that need to be aware of the change.
 - Is an application sending messages to the wrong queue? Check that the messages your application is receiving are not really intended for an application servicing a different queue. If necessary, change your security definitions to prevent unauthorized applications from putting messages onto the wrong queues. If your application has used an alias queue, check that the alias points to the correct queue.
 - Has the trigger information been specified correctly for this queue? Check that your application is the one that should have been started; should a different application have been started?
 - If these checks do not enable you to solve the problem, you should check your application logic, both for the program sending the message, and for the program receiving it.
- ▶ Problems with incorrect output when using distributed queues:

If your application uses distributed queues, you should also consider the following points:

 - Has WebSphere MQ been correctly installed on both the sending and receiving systems, and correctly configured for distributed queuing?
 - Are the links available between the two systems? Check that both systems are available and connected to WebSphere MQ. Check that the connection between the two systems is active. You can use a WebSphere MQ **PING** command against either the queue manager (**PING QMGR**) or the channel (**PING CHANNEL**) to verify that the link is operable.
 - Is triggering set to on in the sending system?
 - Is the message you are waiting for a reply message from a remote system? Check that triggering is activated in the remote system.
 - Is the queue already full? This could mean that an application was unable to put the required message onto the queue. If this is so, check that the message has been put onto the dead-letter queue. The dead-letter queue header contains a reason or feedback code explaining why the message could not be put onto the target queue.

- Is there a mismatch between the sending and receiving queue managers?
- Are the channel definitions of the sending and receiving ends of the channel compatible?
- Is data conversion involved? If the data formats between the sending and receiving applications differ, data conversion is necessary. Automatic conversion occurs when the **MQGET** is issued if the format is recognized as one of the built-in formats. If the data format is not recognized for conversion, the data conversion exit is taken to allow you to perform the translation with your own routines.

10.1.2 Client

An WebSphere MQ client application receives MQRC_* reason codes in the same way as non-client WebSphere MQ applications. However, there are additional reason codes for error conditions associated with clients:

- ▶ Remote machine not responding.
- ▶ Communications line error.
- ▶ Invalid machine address.

The most common time for errors to occur is when an application issues an **MQCONN** or **MQCONNX** and receives the response MQRC_Q_MQR_NOT_AVAILABLE. An error message, written to the client log file, explains the cause of the error. Messages may also be logged at the server depending on the nature of the failure. Also, check that the application on the WebSphere MQ client is linked with the correct library file.

Terminating clients

Even though a client has been terminated, it is still possible for the process at the server to be holding its queues open. Normally, this will only be for a short time, until the communications layer detects that the partner has gone.

Error messages with clients

When an error occurs with a client system, error messages are put into the error files associated with the server, if possible. If an error cannot be placed there, the client code attempts to place the error message in an error log in the root directory of the client machine.

Error messages for UNIX clients are placed in the error logs in the same way as they are for the respective WebSphere MQ server systems. Typically, these files appear in `/var/mqm/errors` on UNIX systems.

Symptom: WebSphere MQ client fails to make a connection

When the WebSphere MQ client issues an **MQCONN** or **MQCONNX** call to a server, socket and port information is exchanged between the WebSphere MQ client and the server. For any exchange of information to take place, there must be a program on the server machine whose role is to listen on the communications line for any activity. If there is no program doing this, or there is one but it is not functioning correctly, the **MQCONN** or **MQCONNX** call fails, and the relevant reason code is returned to the WebSphere MQ application.

If the connection is successful, WebSphere MQ protocol messages are exchanged and further checking takes place. During the WebSphere MQ protocol-checking phase, some aspects are negotiated while others cause the connection to fail. It is not until all these checks are successful that the **MQCONN** or **MQCONNX** call succeeds.

10.1.3 Cluster

This section provides problem determination information about WebSphere MQ clusters.

Overview

When the full WebSphere MQ product is used as the JMS provider, the WebSphere message listener service can be configured to take advantage of WebSphere MQ server clustering.

In the following figure, you can see two WebSphere hosts with a horizontal cluster and a messaging host used to distribute messages for WebSphere MQ server clustering. In this scenario, the listener configurations are the same for each application server.

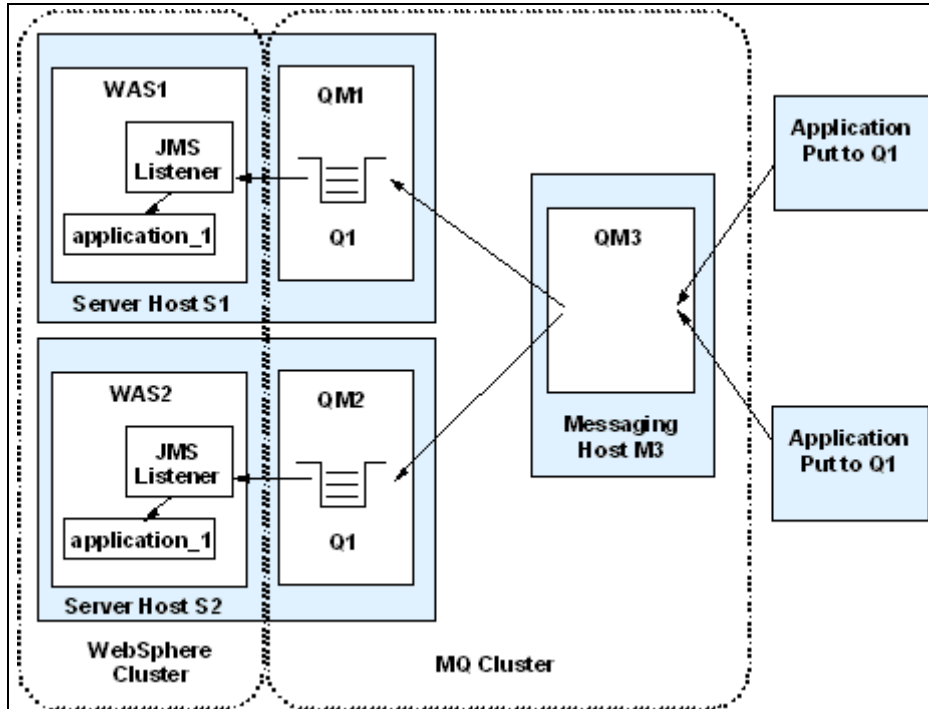


Figure 10-1 Configuration

In the above figure, hosts S1 and S2 contain WebSphere MQ server and WebSphere Application Server.

WebSphere MQ is defined so as to have queue managers QM1 and QM2 and a local queue called Q1 in both the queue managers. The queue managers belong to a cluster. The queue in each queue manager is populated by the WebSphere MQ server located on host M3. Applications can add messages directly to the queue Q1 but will not be subjected to the control of the WebSphere MQ cluster.

WebSphere Application Server contains a member of the horizontal server cluster. Both WAS1 and WAS2 are configured with a JMS listener. The listener is configured to retrieve messages from JMS destination Q1 from the respective queue managers.

Messaging host M3 contains WebSphere MQ server and is defined so as to have a queue manager called QM3, which also belongs to the same cluster as QM1 and QM2. Applications add messages to the queue manager and queue Q1. Because the queue manager belongs to a cluster, the messages are distributed to all other queue managers in the cluster that have Q1 defined. Q1 is not

defined as a local queue on host M3. If the queue was defined locally, then messages will remain on the server for local processing

Symptom: Queue manager fails

If a message-batch is sent to a particular queue manager and that queue manager becomes unavailable, there could be several reasons:

- ▶ With the exception of non-persistent messages on a fast channel (which might be lost), the undelivered batch of messages is backed out to the cluster transmission queue on the sending queue manager.
- ▶ If the backed-out batch of messages is not in doubt and the messages are not bound to the particular queue manager, the workload management routine is called. The workload management routine selects a suitable alternative queue manager and the messages are sent there.
- ▶ Messages that have already been delivered to the queue manager, or are in doubt, or have no suitable alternative, must wait until the original queue manager becomes available again.

Symptom: Repository fails

Cluster information is carried to repositories (whether full or partial) on a local queue called `SYSTEM.CLUSTER.COMMAND.QUEUE`. If this queue fills up, perhaps because the queue manager has stopped working, the cluster information messages are routed to the dead-letter queue. If you observe that this is happening, from the messages on your queue-manager log, you will need to run an application to retrieve the messages from the dead-letter queue and reroute them to the correct destination.

If errors occur on a repository queue manager, you will see messages telling you what error has occurred and how long the queue manager will wait before trying to restart.

In the unlikely event of a queue manager's repository running out of storage, you will see storage allocation errors appearing on your queue-manager log. If this happens, stop and then restart the queue manager. When the queue manager is restarted, more storage is automatically allocated to hold all the repository information.

Symptom: Put-disables a cluster queue

When a cluster queue is put-disabled, this situation is reflected in the repository of each queue manager that is interested in that queue. The workload management algorithm attempts when possible to send messages to destinations that are put-enabled. If there are no put-enabled destinations and no local instance of a queue, an `MQOPEN` call that specified `MQ00_BIND_ON_OPEN` returns a return code of `MQRC_CLUSTER_PUT_INHIBITED` to the application. If

MQ00_BIND_NOT_FIXED is specified, or there is a local instance of the queue, an **MQOPEN** call succeeds but subsequent **MQPUT** calls fail with return code **MQRC_PUT_INHIBITED**.

You may write a user exit program to modify the workload management routines so that messages can be routed to a destination that is put-disabled. If a message arrives at a destination that is put-disabled (because it was in flight at the time the queue became disabled or because a workload exit chose the destination explicitly), the following will happen. The workload management routine at the queue manager may choose another appropriate destination if there is one, or may place the message on the dead-letter queue, or, if there is no dead-letter queue, may return the message to the originator.

Symptom: Repositories lost the information to be retained

When a queue manager sends out some information about itself, for example to advertise the creation of a new queue, the repository queue managers store the information for 30 days. To prevent information in the repositories from expiring, queue managers automatically resend all information about themselves after 27 days. If no update is received within 90 days of the expiry date, the information is removed from the repositories. The period of 90 days is to allow for the fact that a queue manager may have been temporarily out of service. If a queue manager becomes disconnected from a cluster for more than 90 days, it will cease to be part of the cluster at all. However, if it reconnects to the network, it will become part of the cluster again. Note that repositories do not use information that has expired to satisfy new requests from other queue managers.

Similarly, when a queue manager sends a request for up-to-date information from a repository, the request lasts for 30 days. After 27 days, WebSphere MQ checks the request. If it has been referenced during the 27 days, it is remade automatically. If not, it is left to expire and is remade by the queue manager if it is needed again. This is to prevent a build-up of requests for information about dormant queue managers.

Symptom: Issues with cluster resolution

Some common problems with clustering manifest themselves as **MQRC_UNKNOWN_OBJECT_NAME** or something similar, which has at its root the fact that the queue manager to which the application is connected cannot find a reference (either clustered or local) to the required queue. There are several options that can be checked to try and resolve the problem.

- ▶ Use the **amqsput** sample program to try to put a message to the queue. Sometimes this action can cause the queue manager to update its cluster information to find that the queue is available after all.

- ▶ Check that the channels between the queue managers are running correctly. This can be done using the **DIS CHS** command described earlier. If the channels are not running, find the reason why and start them.
- ▶ If clustered alias queues are being used, check that the default binding option on the queue is set to **DEFBIND (NOTFIXED)**. This can be done using the **ALTER QA** command. If this is not set, the queue manager will insert explicit queue manager names into the transmission header, which can break both load balancing and using clustered alias queues.

10.1.4 Channels

Channels are objects that provide a communication path from one queue manager to another. Channels are used in distributed queuing to move messages from one queue manager to another. They shield applications from the underlying communications protocols. The queue managers might exist on the same or a different platform. For queue managers to communicate with one another, you must define one channel object at the queue manager that is to send messages, and another, complementary one, at the queue manager that is to receive them.

Symptom: When the channel refuses to run

- ▶ Check that DQM and the channels have been set up correctly.
- ▶ There may be a mismatch of names between sending and receiving channels (remember that uppercase and lowercase letters are significant).
- ▶ There may be incorrect channel types specified.
- ▶ The sequence number queue (if applicable) is not available, or is damaged.
- ▶ The dead-letter queue is not available.
- ▶ The sequence number wrap value is different on the two channel definitions.
- ▶ A queue manager, CICS system, or communication link is not available.
- ▶ Following a restart, the wrong queue manager may have been attached to CICS.
- ▶ A receiver channel might be in STOPPED state.
- ▶ The connection might not be defined correctly.
- ▶ There might be a problem with the communications software (for example, is TCP running?).
- ▶ In z/OS using CICS, check that the DFHSIT SYSIDNT name of the target CICS system matches the connection name that you have specified for that system.

- ▶ If you get Channel not starting error AMQ9505 with reason code 2085, then review your channel definitions and make sure that the transmission queue specified has a usage of TMQ instead of NORMAL.
- ▶ Check to make sure the client channel definition does not have a blank Queue Manager Name field.

On WebSphere MQ for iSeries™, Windows, UNIX systems, z/OS without CICS, and MQSeries for OS/2 Warp, there is no need for the administrator to choose a particular sequence number to ensure that the sequence numbers are put back in step. When a sender channel starts up after being reset, it informs the receiver that it has been reset and supplies the new sequence number that is to be used by both the sender and receiver. If the sender is WebSphere MQ for z/OS using CICS, the sequence number should be reset to the same number as any receiving queue manager. If the status of a receiver end of the channel is STOPPED, it can be reset by starting the receiver end. This does not start the channel; it merely resets the status. The channel must still be started from the sender end.

Symptom: Build-up of socket files in the /tmp directory

Occasionally, the /tmp directory of the user's application fills up with WebSphere MQ PID files and causes the application and WebSphere MQ to fail. The files are created by MQ channel processes, for example runmqslr, runmqchl, or amqcrsta. They are used for communication purposes, that is, other processes may send commands or requests to the channel process by using these files.

In most cases, these files will be removed automatically. We expect that the files in this case are not being removed because some abrupt method of ending the channels is being used which does not allow the channel process to remove the file, for example, manually killing the process instead of gracefully stopping it. When a UNIX machine is rebooted, files in /tmp are not automatically removed. This has to be done specifically by the user.

Symptom: Problem starting a channel and receiving MSGAMQ9555

AMQ9555 indicates that the AMQRSYNA.DAT file is corrupt. You can correct this in one of the following two ways.

- ▶ You can perform RCRMQMOBJ *ALL *SYNCFILE [QMGRNAME] if you are regularly issuing a RCDMQMIMG *ALL *ALL [QMGRNAME]. OR
- ▶ Quiesce, delete, and recreate your qmanager to create a new AMQRSYNA.DAT file.

Symptom: SNA sender channel may be stuck in retrying status

The SNA sender channel on UNIX may be stuck in retrying status when trying to connect to a mainframe queue manager. If this happens, the AMQ9213 message with return code 24 (X'18) from CPI-C call will be written to the queue manager's error log. There are two possible alternatives for this correction.

- ▶ If the TPNAME and MODENAME are defined for the sender channel, then the CONNAME field should be pointing to a fully qualified partner name.
- ▶ If the CONNAME field is already defined, then the TPNAME and MODENAME *must not* be filled in.

Symptom: Channels will not start after MQ is restarted

Rebooting the server or restarting MQ causes channels (SDR) to fail to come up. AMQ9527, AMQ9511 and AMQ9506 are all generated when the channels try to come up. A possible cause for these errors is a problem with the file system. Most probably, there is not enough disk space. To correct the problem, the channels needed to be deleted and recreated after the file system was modified.

10.1.5 Security

This section provides problem determination details related to WebSphere MQ security.

Symptom: Filling up messages in the dead letter queue

Sometimes, an application does not get started and will start putting the messages in the dead letter queue. The messages in the dead letter queue give the error MQFB_APPL_CANNOT_BE_STARTED.

Typically, an application is started by a call to the `system()` C runtime function. If the launching user ID (which will be the user under which the application will run) does not have at least read access to the ROOT of the drive wherever the application exists then the `start` command will fail with authorization problems. Hence, the user ID with which you start the application should have at least READ access to the root directory which is consistent with normal installation.

Symptom: SSL Channel problem

Whenever you get CSQX720E GSK_Secure_Sock_init fails with -2 status code, it means that there is *no* client or server certificate available. Apart from providing the client and server certificates, you should check to make sure that the certificates in the key store are owned by the user ID, for example, the user ID associated with running the MQServer. Also display the certificate to ensure that it has a private key.

Symptom: Security issues in starting WebSphere MQ

Make sure to set up a domain user ID to delegate authority and also put the domain ID in the global domain mqm group.

Symptom: AMQ9697, WebSphere MQ Services could not be contacted on the target server

There could be a problem setting up SSL to use with a WebSphere MQ client. Set up the SSL so that the local queue managers can use SSL for queue manager to queue manager connections. However, if you try to use WebSphere MQ client from the same machine, the connection will fail and you will get a message regarding the lack of a certificate. Following are two ways you can check this.

- ▶ Each queue manager should have its own private key and a self-signed certificate. This certificate should also act as the CA certificate and at the same time, each key should be exported, then imported into any queue manager's store with which communication is desired. Sometimes, self-signed certificates may lead to SSL related channel failures if you do not have the latest version of GSKIT.
- ▶ Each queue manager should have a private key that has been signed by a certificate authority. For the queue managers to communicate, the private keys must be trusted. This is achieved by checking that the signer of the certificate is a trusted authority. The CA certificate used to sign the private certificate must be present in the receiving queue manager.

Symptom: Changing permissions on var/mqm causing client connection problems

In a Unix environment, changing the all access permissions for others on the /var/mqm and /opt/mqm directories will cause the client connections to fail with an mqrc=2059 on the MQCONN. AMQ9503 (Channel negotiation fails) and AMQ9228 (TCPIP responder program could not be started) can be found in the log files.

You should never change the permissions of /var/mqm and /opt/mqm. The file permission on the /opt/mqm directory should be at least 555 (r-xr-xr-x) and the file permissions on /var/mqm should be at least 775 (rwxrwxr-x). This is because users outside of the mqm group can run WebSphere MQ applications which will need to be able to access files within /var/mqm such as /var/mqm/mqs.ini, /var/mqm/qmgrs/QmgrName/@ipcc/*. Without this fundamental UNIX permission to access these files, users will not be able to access the required files.

Symptom: MQJMS2013 Invalid Security Authentication

An application failing with MQJMS2013: invalid security authentication supplied for MQQueueManager error only occurs when attempting to connect in bindings mode. If you are using client mode to connect, there is no problem.

The reason for this error could be not having the sufficient authorities. Check which user id the application is being run under and then check to see if that user id is in the mqm group. If it is not, then add it to the mqm group and issue a `runmqsc REFRESH SECURITY(*)` command.

Symptom: Dead Letter Handler failing with reason code 2035

If you have many messages which went to dead letter queue on WebSphere MQ Queue manager, then running the dead letter handler might fail with 2035 reason code.

The reason could be that the messages were trying to use the user ID in the message descriptor, that is, the messages in the dead letter queue contain several different IDs and none of those has been defined on your platform. This usually happens if you specify PUTAUT (CTX) for the dead letter handler in the rules table. The solution is remove this from the rules table and let it default to PUTAUT (DEF) so that it uses the ID under which the runmqdlq was running.



WebSphere Business Integration Message Broker

This chapter covers WebSphere Business Integration Message Broker. It details important things to check, explains the different types of tracing and how to use the Workbench's debugger. There is also a list of common symptoms together with their associated problems.

11.1 WebSphere Business Integration Message Broker

WebSphere Business Integration Message Broker is a very complex product which relies upon other products and requires appropriate security permissions to perform all the tasks.

The infrastructure of WebSphere Business Integration Message Broker is as follows. The broker uses a DB2 database (which can be local or remote) to store information pertaining to the message flows, configuration, runtime resources, etc. The flows themselves can also interact with databases, DB2 or otherwise, as part of the application. Each broker is associated with a WebSphere MQ queue manager; this is the only queue manager that the broker uses to retrieve and send messages. The broker is controlled via the Configuration Manager, which runs on the Windows platform only. The Configuration Manager has a DB2 database and WebSphere MQ queue manager of its own (if the broker and Configuration Manager reside on the same machine, they can share a queue manager). The Configuration Manager is then controlled using the Message Brokers Toolkit for WebSphere Studio.

All these different products need to be correctly configured and started in order for the system to function correctly. With such a complex setup, problems are almost inevitable, but hopefully you will be able to find these problems with the aid of this chapter.

Things to check

- ▶ First, have the main products Business Integration Message Broker replies to all been started successfully?
 - Is the WebSphere MQ queue manager running?
 - Is DB2 started, including any required remote databases?
- ▶ Has WebSphere Business Integration Message Broker ever run successfully before?

If not, it is likely that there is something wrong with the installation or setup. Refer to the installation guide for more information.

It is important that the sample profile, found in the `install_dir/sample/profiles` directory, be customized and included in the profile of any user hoping to run broker commands.

- ▶ Did you log out of Windows while WebSphere Business Integration Message Broker components were active?

In Windows, logging off while components are active can cause problems. You might see messages including 2070, 2642, 1102, and 1103 in the Windows Event Log if this is the case.

To avoid this problem, set the queue managers used by the broker, User Name Server and Configuration Manager, to run as Windows services, so that logging off Windows does not cause this problem.

- ▶ Are the UNIX environment variables set correctly?

Providing the sample profile is used and set correctly, this should not be a problem.

Ensure the following variables are defined correctly: CLASSPATH, LANG, LC_ALL, LD_LIBRARY_PATH, LC_MESSAGES, MQSI_REGISTRY, ODBC_INI, NLSPATH and PATHUse. See the product installation guide for more information.

- ▶ Are there any error messages or return codes that explain the problem?

On UNIX systems, if the syslog daemon is configured (see “Syslog daemon” on page 51), error messages are written to the file `/var/adm/user.log`. Messages from WebSphere Business Integration Message Broker have MQSIv500 in the message, for example:

```
Dec 5 11:52:38 rs617001 MQSIv500[37508]: (LGI_BROKER_ITS0_01)[1]BIP2001I:
The WebSphere Business Integration Message Brokers service has started,
process ID 55662. : LGI_BROKER_ITS0_01.service:
/build/S000_P/src/AdminAgent/ControlProcess/riios_aix_4/ImbControlService.cp
p: 337: ImbControlService::StartNewAA: :
```

- ▶ Can you reproduce the problem?

If so, then depending upon how it is reproduced, different steps should be taken.

- If the problem is caused by a particular message flow then use the debugger facility of the Toolkit, as covered in 4.5.2, “Message Broker Debugger” on page 90.
- If the problem is caused by a command, then check whether it occurs when executed from other accounts. Does the user’s profile include a correctly customized sample profile, and are all variables correct and complete?

- ▶ Has the message flow run successfully before?

If the problem seems to focus around one message flow, consider whether it has been *successfully* run previously. Before answering this positively, consider the following:

- Has the flow been changed since it was successfully run?

In this case, it is very likely that the cause of the problem lies in the changed areas of the flow. Check that they are correct, using the debugger if required.

- Has the full functionality of the flow been used before?

Perhaps the problem is in a rarely used part of the flow which has never actually worked. Use tracing or the debugger to determine if this is the case.

- Does the flow check for all possible return codes, and how does it handle errors?

It is possible that due to changes made elsewhere in the system, the array of possible return codes has been enlarged, and the flows have not been adjusted to allow for this.

Check the error handling carefully and ensure that it is backing up updates or not, as required, and that messages are being put out where they are expected and are not looping.

- Does the message flow expect particular message formats?

If a message with an unexpected message format has been put onto a queue (for example, a message from a queue manager on a different platform), it may require data conversion or some form of processing.

- Does the message flow run on other WebSphere Business Integration Message Broker systems?

Perhaps there is something which is specific to your setup which is causing the problem. Running the flow on another system could help determine if the flow or the setup is at fault. Errors could include database errors or problems with LIL files. These may be incorrectly compiled, or perhaps the broker does not have permission to access them.

- ▶ Have you made any changes since the last successful run?

Don't just consider changes to WebSphere Business Integration Message Broker, but also to other parts of the system. Has any software been upgraded to newer levels or versions? On Unix, has the profile of the user which the commands are run on been altered? Are there other modifications to be taken into account?

- ▶ Is there a problem with a command?

On UNIX platforms, be careful when including special characters, for example, backslash (\) and double quote (") characters, in descriptive text for some commands. If you use either of these characters in descriptive text, precede them with a \, that is, enter \\ or \" if you want \ or " in your text, respectively.

► Is there a problem with a database?

If there is a database problem, perform the following checks:

- Ensure the database is started; type `db2start` from the correct user account. If it is already started, you will be told so.
- On Windows, check that there is an ODBC connection for each database defined on the System DSN tab of Data Sources.
- On UNIX, check the `odbc.ini` file (by default, `/var/mqsi/odbc/.odbc.ini`), which contains the correct database definition. For the broker database, the definition is correct if the broker can be created. Use ODBC tracing if the broker cannot be started.
- Check the number of database connections: on DB2 for AIX, using local mode connections, the maximum is 10. An alternative to local mode is to use TCP/IP.
- If the database is a remote database that is being accessed from a message flow, ensure that the database `mqsi setdbparms` command has been run correctly and all associated configuration completed.
- If the database is being accessed from a message flow, check that the message flow node is showing the correct information for the data source and table names.
- Records or groups of records can be locked by DB2 until updating is complete. If many applications are using a database, it could be a reason for occasional failure.

► Is there a problem with the network?

WebSphere Business Integration Message Broker relies on WebSphere MQ as its communication medium. If two components are on separate queue managers, these are connected using message channels. Ensure all required channels are defined and started.

If the two queue managers are on separate machines and the channels still will not start, try `ping <hostname>` to see if there is a problem with the underlying network.

Setting up channels so that they are triggered is good practice because it prevents them from timing out.

► Does the problem affect all users?

Some users may have their environments set correctly for running WebSphere Business Integration Message Broker while others may not. Ensure that all WebSphere Business Integration Message Broker commands are running under the correct user. If not, stop running components using their current user and restart them with the correct user.

On AIX, always be careful to change the profile as well as the user.

- ▶ Have you recently changed a password?

If operating system passwords have been changed for the user accounts that are used for either database access or WebSphere Business Integration Message Broker components, then WebSphere Business Integration Message Broker needs to be aware of these new passwords. Use `mqsichangebroker`, `mqsichangeconfigmgr` or `mqsichangeusername` to set this data.

Changing passwords or user IDs can also affect WebSphere MQ, so ensure that all necessary steps covered in Chapter 8 of the *WebSphere MQ System Administration Guide* have been followed.

- ▶ Have you applied any service updates?

If any CSDs or interim fixes have been applied, check that no error messages were produced. If the installation was successful, check the IBM Support Center to see if there are any known errors.

Ensure that all instructions for the application of the CSD were followed exactly, and that no steps were missed. Do resources need redefining, for example?

To check whether any updates have been applied to WebSphere Business Integration Message Broker, check the `memo.prt` file, usually located in the `/usr/opt/mqsi/readmes/En_US` on UNIX.

Note: Also check that you have the service levels that are correct for dependencies. For example, CSD02 of Message Brokers V5 requires FP3 of DB2 8.1

If an update was applied to any other products or connected systems, consider the effect this may have.

- ▶ Do you have a component that is running slowly?

If a particular component, or the system in general, is running slowly, try the following:

- a. Check whether you have inadvertently left tracing on. This could be WebSphere Business Integration Message Broker user tracing or service tracing, ODBC tracing, WebSphere MQ tracing, or native database tracing. If you have, turn the tracing off.
- b. Clear out any old abend files from your errors directory. If you do not clear it of unwanted files, you might find that your system performance degrades due to significant space being used up.

- c. On Windows, use the work path `-w` flag of the `mqsicreatebroker` command to create the errors directory in a hard drive partition that does not contain WebSphere Business Integration Message Broker or Windows itself.
- d. Increase your system memory.

This section provides detailed information about WebSphere Business Integration Message Broker problem determination.

Symptom: Broker fails to start

If the problem is that the broker has failed to start, it is, first of all, important to understand how to determine that it has not started. Example 11-1 shows the screen output of an unsuccessful start. It is identical to the screen output of a successful start. It should, however, be evident from the workbench that the broker has not started. The syslog file (see the link to syslog in the tools), `/var/adm/user.log`, should indicate what the problem is.

Example 11-1 Screen output of an (un)successful starting of broker

```
$ mqsistart LGI_BROKER_ITSO_01
WebSphere MQ queue manager running.
BIP8096I: Successful command initiation, check the system log to ensure that
the component started without problem and that it continues to run without
problem.
```

There are many reasons why a broker may fail to successfully start. The two major dependencies are the broker's database and queue manager. These must be started, and the broker must have the correct authorization for access.

Example 11-2 syslog of broker start failure due to incorrect authorization credentials (edited for clarity)

```
The WebSphere Business Integration Message Brokers service has started, process ID 39134.
An Exception was caught while issuing database SQL command connect.
Database error: ODBC return code '-1'.
Database error: SQL State '08001'; Native Error Code '-30082'; Error Text '[IBM][CLI Driver]
SQL30082N Attempt to establish connection failed with security reason "24" ("USERNAME
AND/OR PASSWORD INVALID").
The broker made an unsuccessful attempt to access its database BROKERDB with userid db2inst1.
```

Example 11-3 on page 288 shows the syslog output that is generated when the broker is attempting to access its database with the wrong password. It is quite clear what the problem is here and how to fix it. The command `mqsichangebroker -p password` sets the password the broker will use to access its datasource. The username cannot be changed from the one set initially.

Example 11-3 syslog of broker start failure due to database not being started

The WebSphere Business Integration Message Brokers service has started, process ID 67690. An Exception was caught while issuing database SQL command connect.
Database error: ODBC return code '-1'.
Database error: SQL State '08001'; Native Error Code '-30081'; Error Text '[IBM] [CLI Driver] SQL30081N A communication error has been detected. Communication protocol being used: "TCP/IP". Communication API being used: "SOCKETS". Location where the error was detected: "". Communication function detecting the error: "connect". Protocol specific error code(s): "79", "*", "*". SQLSTATE=08001.
The broker made an unsuccessful attempt to access its database BROKERDB with userid db2inst1.

Example 11-3 shows the syslog when the database is not running. Unfortunately, it is not perfectly clear that this is the problem; indeed, the log simply says that there is a communications failure. In this case, the database is accessed by TCP/IP, although it is local. First, ensure that DB2 is started; if it is, there is a more serious error. Is it possible to connect from the DB2 command line? Checking the db2diag file (see “db2diag.log file” on page 95), could lead to more clues.

Symptom: Broker starts, but flows containing database access fails on AIX

By default, AIX uses System V shared memory in a way that is different from other UNIX platforms. This results in a limit of ten shared memory segments being attached by a single process. Using EXTSHM allows more than ten segments to be attached by a single process. This makes AIX behavior similar to the behavior of other UNIX platforms, but there is still a limit. With WebSphere MQ, DB2, and WebSphere Business Integration Message Broker all on one AIX system, these can soon run out. Example 11-4 shows a db2diag.log entry when a flow could not access the database it required.

Example 11-4 db2diag logfile entry for shared memory failure

```
2003-12-11-11.39.31.089907 Instance:db2inst1 Node:000
PID:67968(DataFlowEngine) TID:7711 Appid:none
SQO Memory Management sqlocshr2 Probe:200

pSetHdl:
0x3B5417A0 : FFFF FFFF FFFF FFFF 0000 0000 0018 002B   yyyyyy.....+
0x3B5417B0 : 0004 0000                                     ....
```

Note that the example shows the process attempting the access, DataFlowEngine. The relevant information is contained in the third line. The problem occurred in the SQO Memory Management component in the sqlocshr2 function. Probe:200 indicates the position within the function where the error

occurred but is irrelevant here. From these two pieces of information, you can make an informed guess that shared memory is the problem.

Solution: The solution to this problem is to access databases by TCP/IP instead of using AIX's shared memory. In some cases, this has also been known to improve performance. To do this, the database must be recatalogued to a node. This may require you to recreate the broker and its database, as there can be problems cataloguing the database remotely with the same name and it is not possible to change the name of the database the broker tries to access.

Here are the steps that can be followed when resolving this issue:

1. Delete the broker.
 - a. As the correct user, run `mqsistop <BROKER_name>`
 - b. Delete the broker: `mqsdelete <BROKER_name>`
 - c. Using the DB2 user ID, for example: `db2inst1`, drop the broker's database
2. Create a database, with a name different from that of the alias which you wish to create, for example, if the database alias is to be BROKERDB, name the database BROKERL (for local).
3. Catalog a node at the DB2 command line type: `catalog TCPIP node <NODENAME> remote <HOSTNAME> server <SERVICENAME>`
To determine the correct port number to use, type: `get dbm cfg` (at the DB2 prompt) and find the value for SVCENAME.
4. Catalog the database as an alias on the new node; at the DB2 command prompt type, `catalog db <DATABASE> as <ALIAS> at node <NODENAME>`.
5. Ensure the entry in the `odbc.ini` file correctly points to the alias name for the database.
6. Restart DB2.
7. Check that it is possible to connect the database, with a command like:
`db2 connect to BROKERDB user db2inst1 using db2inst1`
Providing this works, the database is now ready to be used and you can now recreate your broker.

Symptom: Unable to attach workbench debugger to broker

The most likely problem here is that the Agent Controller is not running on the broker machine. To determine if it is running, type: `ps -ef | grep RAServer` on the broker machine. If the result is as in Example 11-5 on page 290 then the agent is running. If nothing is outputted then the agent is not started. Start it by typing `/usr/IBMRAC/bin/RAstart.sh` at the command prompt. If it fails to start, simply try again.

Example 11-5 Example showing the Agent Controller running

```
$ ps -ef | grep RAServer
root 68700      1  0 17:45:37      -  0:00 RAServer
```

If starting the Agent Controller did not fix the problem, or the Agent Controller was already started, then try stopping and starting the broker. It seems the workbench cannot find the broker's process if it was started before the Agent Controller.

If the problem still persists, consider the possibility that there is a network problem. Is the hostname entered correctly in the workbench? Is it possible to ping the broker machine from the workbench machine? Is there a firewall between the two machines? The firewall may be blocking the port.

Symptom: Error BIP2066E while attempting to deploy flows to a UNIX broker

Check the value of the environment variable LANG on the broker machine. It should be C for the U.K. and En_US for the United States.

Symptom: Error BIP8053E when attempting to create a Configuration Manager

Is the user trying to perform the action as a member of the required mqbr* groups? The WebSphere Business Integration Message Broker V5 installation no longer creates all the required groups and users automatically. On Windows, the Security Wizard can be started from the Start menu and used to create the groups and put users in them.

Solution: Create the required groups and add the required user(s) to these groups. See “Setting up broker domain security” in the Software Information Center for more information.

Symptom: Compute node cannot access elements of a delimited message set

If you are having problems with a Compute node accessing the elements of a delimited message set (for example with a colon delimiter string1:string2) then check that you have set the delimiter to ':' and set the type to All Elements Delimited. If you are not using the RFH2 header to identify the incoming message, then you will need to set the Message Domain, Set, Type and format in the Default tab of the MQInput node.

Symptom: “Cannot find com.ibm.broker.plugin” error

If you are working with plug-in nodes and you see errors such as Cannot find com.ibm.broker.plugin or Unable to load message catalog - mqji then check that you have added all the required files to your classpath. For example, add jplugin.jar or mqji.properties to your classpath to resolve the errors above, respectively. These files can be found in the classes sub-directory under the product installation directory.

If you have problems with scaling a configuration, processing large messages or processing high volumes of messages then it is a good idea to refer to the performance reports for advice. These are available as support packs.

Symptom: “ConfigManagerProxy information was not received from the Configuration Manager” error

You get the following error when trying to connect to the Configuration Manager from the toolkit.

If ConfigManagerProxy information was not received from the Configuration Manager, either the Configuration Manager is not available or the user 'wbimb@p3028754' does not have authority to view the object. (UUID=", required attribute='name').

This error is stating that the toolkit sent a message to the Config Manager's input queue but the Config Manager did not reply.

Some things to check:

- ▶ Check the status of the Config Manager in the Windows Services panel.
- ▶ Look in the Windows Event log for errors (if your most recent BIP message has the ID "1003" then this means that the Config Manager started successfully).
- ▶ Look at the Config Manager's Queue Manager's SYSTEM.BROKER.CONFIG.QUEUE for messages (this is the Config Manager's input queue). If there are unread messages, the Config Manager is not running.
- ▶ Check in the toolkit that the domain connection details you have entered correspond to the Config Manager that is running. Finally, make sure that the timeout period (the time the toolkit waits for the Config Manager to reply to messages) is set to a suitably large value (in the toolkit, select **Window -> Preferences -> Broker Administration -> Configuration Manager Proxy**. The top three text fields describe the retry timeouts. Example entries would be 5, 1500 and 2000.).

Symptom: Message BIP1511E is issued when creating a broker scenario

The following error message is displayed when you try to create a new broker in the Topology editor:

```
BIP1511E: Queue manager QM1 cannot be associated with broker BR1; it is
already associated with broker BR1
```

The problem has occurred because you have performed an incorrect sequence of actions when you deleted the broker, and you are trying to recreate a broker of the same name.

You must force the deletion of the broker before attempting to recreate it, as follows:

1. Stop the broker using the `mqsistop` command.
2. Delete the broker using the `mqsdeletebroker` command.
3. Delete the broker from the topology.
4. Deploy the changed topology.
5. Check that the deploy was successful, and that the broker has disappeared from the workbench.

Do not to recreate the broker until the deleted topology has been successfully deployed. If the broker is no longer shown in the workbench, you can recreate the broker, as follows:

1. Create the broker using the `mqscreatebroker` command.
2. Create the broker in the topology.
3. Deploy the topology.

Symptom: Error message BIP8075E is issued when creating a Configuration Manager Scenario

On a Windows system, the `mqscreateconfigmgr` command fails with message BIP8075E and a Java exception `__unsatisfiedLinkException__`.

The Configuration Manager cannot find the JAR files that it needs to connect to the configuration database. If you have installed additional software since installing the broker, or have made manual updates to your CLASSPATH, these might cause the `mqscreateconfigmgr` command to fail.

Solution: Ensure that the DB2 JAR files are in your CLASSPATH, especially `db2java.zip`.

Symptom: You do not know what authorities are set as part of the `mqsicreateaclgroup` command

Use the WebSphere MQ `dspmqaout` command to check which authorities have been set on a queue manager by the `mqsicreateaclgroup` command. The following authorities should be set: `inq`, `set`, `connect`, `altusr`, `chg`, `dsp`, `setall`.

Symptom: Creating a domain connection fails with error BIP0889E scenario

When you create a domain connection, it fails with the following error:

```
BIP0889E An exception occurred while communicating with Configuration
Manager on <Queue Manager>. Reason: WebSphere studio is experiencing
problems communicating with the Configuration Manager on <Queue Manager>
Ensure that the Configuration Manager is available If so and exception
still occurs, retry the operation after increasing Configuration Manager
proxy retry attempts value in preferences
```

Details similar to the following are given:

```
PubSubTopology information was not received from the Configuration Manager.
Either the Configuration Manager is not available or the user <username>
does not have authority to view the object. (UUID=PubSubTopology, required
attribute=subcomponent.last)
```

The problem may be that the WebScale Distribution Hub (disthub), which is installed as part of a full WebSphere Business Integration Message Broker installation, has not installed correctly. To verify that disthub has installed correctly, look for the classes and lib folders in the install directory.

There are other reasons why this message may appear and you should also check that the Configuration Manager is running, and that security is set up correctly.

Check that the disthub install directory `C:\Program Files\IBM\Disthub\v2` contains folders called `classes` and `lib`. If the directory contains only an `_uninstall` folder, you need to uninstall disthub, and install it again, as described below:

1. At a command prompt, change to `C:\Program Files\IBM\Disthub\v2_uninst`.
2. Execute `java -jar uninstall.jar`.
3. To complete the uninstallation, select **1,1** and **3**. You can ignore the error messages.
4. The product install package contains a folder called `disthub`. At a command prompt, change this folder.
5. Execute `java -jar setupwin.jar`.

6. Select the options **1,1** and **3** to complete the installation of disthub.

If this does not solve the problem, it may be that the problem is caused by the toolkit user not having access to a remote Configuration Manager. Check that the correct parameters were selected when creating the Configuration Manager and recreate it if necessary. Check that the ACLs are set up correctly.

Symptom: The Getting Started wizard does not appear to have worked

You have run the Getting Started wizard to create a default configuration, but it does not appear to have worked.

The Getting Started wizard creates a WebSphere MQ listener and starts it on the port you specify in the wizard. The wizard currently has no mechanism to check whether a listener already exists on the queue manager, nor whether it is listening on the same port. Therefore, if you are using the Getting Started Wizard to create components on an existing queue manager, you might have a problem.

Check your queue manager, and delete any duplicate listeners. Ensure that the remaining listener is configured to listen to the correct port.

Symptom: Broker fails to start on z/OS

You get abend code 047 and a diagnostic message. On z/OS, your broker gets abend code 047 when you try to start it, and you get the diagnostic:

```
IEA995I SYMPTOM DUMP OUTPUT 463 SYSTEM COMPLETION CODE=047 TIME=10.53.47
SEQ=00419 CPU=0000 ASID=008E PSW AT TIME OF ERROR 078D0000 98D09E52 ILC 2
INTC 6B ACTIVE LOAD MODULE ADDRESS=18D08828 OFFSET=0000162A
NAME=SPECIALNAME 61819987 968995A2 A3618499 89A58599 */argoinst/driver*
F1F46D82 96858261 A4A29961 93979761 *14_boeb/usr/lpp/* A6949889 61828995
61828997 89948189 *wmqi/bin/bipimai* 95 *n * DATA AT PSW 18D09E4C -
58109948 0A6B5820 B8E95020 GPR 0-3 00000000 0000003C 00000000 00000000 GPR
4-7 18D10300 18D115F0 00000013 00000004 GPR 8-11 18D111CF 18D101D0 18D0BBBE
18D0ABBF GPR 12-15 98D09BC0 18D101D0 98D09E22 00000000 END OF SYMPTOM DUMP
```

The previous sample text is written to the SDSF SYSLOG.

System completion code 047 means that an unauthorized program issued a restricted Supervisor Call (SVC) instruction. The diagnostic also indicates that the program in error was bipimain. When installing WebSphere Business Integration Message Broker, you must issue the command **extattr +a bipimain** from the bin directory of the installation path to give program bipimain APF authorization.

Symptom: You cannot start the Configuration Manager

You have created the Configuration Manager specifying the local system name for the Windows security domain, and used the same identifier for the service user ID. When you start the Configuration Manager, the following message is generated: BIP8026W: Unable to start the component. The component could not be started using the service user ID that was supplied when the component was created. This also happens with the User Name Server when you run that on Windows.

The configuration you have specified is not supported by Windows. When you start the component created with this combination of parameters, a Windows system call incorrectly retrieves the SID (Security IDentifier) of the machine rather than that of the account.

If you want to use your local account domain as the domain from which users and groups are drawn, you must specify a different user ID for the service user ID. For example, if you specify `NTSecurityDomainName` on the `-d` flag on the `mqsicreatebroker` command as `NYBROKER` on local system `\\NYBROKER`, the service user ID on the `-i` flag on the `mqsicreatebroker` command cannot be `nybroker`. You must delete your Configuration Manager (or User Name Server, or both) and recreate it using another user ID that has the correct authorizations.

Symptom: You cannot stop the broker

You issue the `mqsistop` command to stop the broker, but the system freezes and never actually stops any of the execution groups.

One possible cause of this problem is that you or someone else is debugging a flow and it is currently stopped at a breakpoint. WebSphere Business Integration Message Broker regards this as a “message in flight” situation, and refuses to stop the broker through the normal command.

The best action is to click **Stop debugging** from the workbench. After that operation has completed, the broker stops. If it is not possible to stop debugging, end all execution group processes associated with that broker to allow the broker to stop, but keep in mind that your messages are backed out and that you still have to click **Stop debugging** after the broker restarts.

Symptom: You cannot delete your broker after deleting your broker database

You deleted your broker database, and now you cannot delete your broker, but get error message BIP8040W.

The **mqsdeletebroker** command checks for the broker database tables and produces this error because the database is not there.

Solution: You can work around this problem by creating a dummy database with the same name as the database you deleted. You must also recreate the ODBC connection. Reissue **mqsdeletebroker**, and then delete the dummy database.

Symptom: The workbench is deploying to a deleted broker

You have deleted your broker using the **mqsdeletebroker** command, and deleted it from the topology, but the workbench is still deploying to the deleted broker.

The problem has occurred because you have either performed an incorrect sequence of actions when you deleted the broker, or you have recreated a broker of the same name before the previous deletion has completed.

To delete the broker correctly, carry out the following procedure:

1. Stop the broker using the **mqsistop** command.
2. Delete the broker using the **mqsdeletebroker** command.
3. Delete the broker from the topology.
4. Deploy the topology.
5. Check that the deploy was successful.

The broker should now have disappeared from the workbench. Do not try to recreate a broker using the **mqscreatebroker** command until you are confident that it has been removed from the workbench. By the same token, do not reuse broker names or queue manager names until you are confident that they are not in use elsewhere.

Symptom: The product hangs when pasting ESQL statements from Adobe Acrobat

When you copy and paste certain ESQL statements from Adobe Acrobat into the ESQL editor, WebSphere Business Integration Message Broker stops responding.

This is a known problem that occurs when pasting text from Adobe Acrobat into the ESQL editor and also into the Java Editor.

To work around this problem, you will have to either manually enter the text, or copy and paste to a text editor like Notepad and perform another copy and paste action from there.

Symptom: You get warnings or errors for message references

You get warnings or errors for message references, and you are sure that your references are correct.

This will never be the case with messages using the XML parser. For these message references, direct validation is not performed because the references could be used for “generic” XML.

To use the validation feature:

1. Create a reference to the tree and parser in the module main procedure.
2. Associate the reference to the correlation name, for example `InputRoot`, `Root`, or create the `OutputRoot.parser` node, where *parser* is the name of the parser that you want to use.
3. -Pass the reference as a parameter to an ESQL subroutine that identifies the XSD type of the reference. This practice is beneficial because the passed reference supports content assistance and validation for ESQL. The message type content properties open or open defined are not used in validation, and the assumption is that this property is closed. There is an ESQL editor preference where you can choose to ignore message reference mismatches, or to have them reported as a warning or an error. By default, this type of problem is reported as a warning, so you can still deploy the message flow.

Symptom: Execution group is not reading messages from the input queues

Your execution group is started, but it is not reading messages from the specified input queues.

A started execution group might not read messages from the input queues of the message flows because previous errors might have left the queue manager in an inconsistent state.

To solve the problem, take the following actions:

1. Stop the broker, the WebSphere MQ listener, the WebSphere MQ channel initiator and the WebSphere MQ queue manager.
2. Restart the queue manager, the channel initiator, the listener, and the broker.

Symptom: You rename a flow that contains errors, but the task list entries remain

When you rename a message flow in the workbench for which there are error icons (red crosses) displayed on nodes and connections, those error icons are removed when changes are made. However, the task list entries remain.

If the message flow editor does not refresh correctly, close and reopen the editor.

Symptom: You cannot deploy a message flow that uses a plug-in message flow

You have created a message flow containing an input node in a user-defined node project. However, you cannot deploy a message flow that uses this plug-in message flow.

The validation, compiling, and deploying functionality cannot currently recognize a plug-in message flow as containing an input node.

To work around the problem, add a dummy input node to the flow you intend to deploy.

Symptom: The message flow deploys on the test system, but not elsewhere

The message flow you have developed deploys on the test system, but not elsewhere, and you need information to help diagnose the error.

Carry out the following checks:

1. Make sure that you have verified the installation by creating and starting a broker, and deploying a single execution group. This shows that WebSphere Business Integration Message Broker and the broker database are in place.
2. Ensure that the broker archive (BAR) file's broker.xml file contains references to the correct resources for the new system.
3. Ensure that any referenced message sets are deployed.
4. If database resources or user-defined nodes are not accessible or authorized from the target system, the deploy fails. On the distributed platforms, ensure that your databases are defined as ODBC sources so that they can be accessed from WebSphere Business Integration Message Broker. Also, you must set the broker's environment to allow access to the databases (you might need to run a profile on UNIX platforms).

5. Any user-defined extensions that you are using in your message flow might not load if they cannot be found, or are not linked correctly. Consult the documentation for your platform for details about tools that can help you check your user-defined extension binaries.

Symptom: You get a correlation name error when deploying to a V2.1 broker

When attempting to deploy a broker archive file to a V2.1 broker, you get the following error message:

```
The correlation name CREATE is not valid. Those in scope are: Environment,
InputLocalEnvironment, OutputLocalEnvironment, InputRoot, InputBody,
InputProperties, OutputRoot, InputExceptionList, OutputExceptionList,
InputDestinationList, OutputDestinationList.
```

This error message may appear because you have attempted to deploy a message flow that contains a mapping node to a V2.1 broker. Mapping nodes are not supported on V2.1 brokers. This can also occur if you are deploying a message flow that is in the same message flow project as another message flow that contains a mapping node to a V2.1 broker. If this is the case, move the message flow that contains a mapping node into a different project, and try the deployment again.

A similar error is thrown if you have not compiled the ESQL and built the BAR file under the correct compatibility level in the ESQL and Mapping Code Generation preferences.

If you are trying to deploy large message sets and you see BIP errors, see the *Problem Determination Guide* “Problems when deploying message flows or message sets” section for guidance.

Symptom: You do not receive confirmation that the deployment was successful

You have initiated a deploy, but have not received a confirmation that the deployment was successful.

Assume that communications from the workbench to the Configuration Manager are working, because the workbench shows an error immediately if this link fails. Follow these steps to find out if there is a communication problem which stops the receipt of the deployment confirmation.

1. Stop the broker using the `mqsistop` command. On the broker queue manager, check the `SYSTEM.BROKER.ADMIN.QUEUE`. If there are messages on this

queue, there might be a broker error; look in the local error log on the broker's system. If the queue is empty, continue with the next step.

2. Redeploy your message flow. Recheck the SYSTEM.BROKER.ADMIN.QUEUE on the broker queue manager. If the queue depth has not increased, there is a problem with the channel between the Configuration Manager and the broker; check the WebSphere MQ logs. If the queue depth has increased to one, continue with the next step.
3. Stop the Configuration Manager, and then start the broker. This allows the broker to process the configuration change and send a response. Wait for the broker to process the message: this could take some time. If the message takes more than five minutes to process, there is a problem with the deployed message and a set of failure messages appear in the local error log on the broker's system indicating the reason for the failure. Check the SYSTEM.BROKER.ADMIN.REPLY queue on the Configuration Manager's queue manager. This should contain a response. If it does not, check that the channel from the broker to the Configuration Manager is running.
4. Restart the Configuration Manager. Check that the messages are read from the SYSTEM.BROKER.ADMIN.REPLY queue on the Configuration Manager's queue manager. If they are not, check the local error log on the Configuration Manager's system for errors.
5. Refresh the Event Log editor in the workbench. You should now get a reply.

Symptom: Other users on the Toolkit machine are able to access your workspace

The workspace is in the default location in the directory tree and available to other users. You would prefer the workspace directory for the workbench to be located under your documents and settings, rather than under the install path.

You can edit the properties of the shortcut to mqsistudio.exe in the Start menu, or create a new shortcut to mqsistudio.exe, and edit the properties for this shortcut. To edit the properties of the new or existing shortcut, under the Shortcut tab, there is an option (Start in:), which indicates the location for the workspace. Enter a new path there. This path can be to a remote machine, or to local or network drives. If you already have a workspace you want to keep, you can copy this existing workspace to the new location.

Alternatively, you can enter the following string under Windows **Start -> Run**:
"C:\Program Files\IBM\WBIMB\eclipse\mqsistudio.exe" -data "C:\Documents and Settings\user1\Workspacel\mqsistudio.exe".

If you have problems with Publish/Subscribe, see the section on problems when using publish/subscribe in the product problem determination guide.



The back-end diagnostic overview

From a review of the topology diagram related to the eMerge application, we can see that the key components that are unique to the back-end are:

1. TXSeries on AIX: while the primary component, as far as the application is concerned, is CICS, Encina® is used to manage the file system, and DCE might be used to manage the security functions. Each of these components has specific diagnostic procedures and sources of data.
2. CICS Transaction Gateway on AIX.
3. CICS Transaction Gateway on z/OS.
4. CICS/TS on z/OS.

We will discuss some of the key sources of diagnostic data for each of these components and the tracing options that can be used to assist with application related problem diagnosis.

12.1 WebSphere TXSeries CICS

The key pieces of diagnostic information required for diagnosing TXSeries/CICS problems include not only CICS specific data, but also data related to the Encina and DCE components of TXSeries.

In WebSphere TXSeries, the CICS component basically provides a CICS API interface. The transaction coordination and management process runs as a global task under Encina. Encina is the transaction manager, the two-phase commit coordinator and controls access to the Structured File Server (SFS). Even if DB2 is used as the file system, Encina will be the two-phase commit coordinator.

Most installations that run WebSphere TXSeries CICS use Remote Procedure Calls (RPC). This does not use DCE for authorization. DCE in this case is merely the portal through which connections are passed via RPC to CICS.

The recommendation, in the early stages of problem diagnosis, is to capture a full CICS trace, and if a problem with the SFS is also indicated, then Encina tracing should also be active. Additional, more granular, trace data may be required to progress the diagnosis.

12.1.1 The SYMREC file

When the CICS region writes a symptom record, the record is appended to the file `/var/cics_regions/regionName/symrecs.nnnnnn` (on CICS for Open Systems) or `c:\var\cics_regions\regionName\symrecs.nnnnnn` (on CICS for Windows NT®). This file does not exist until a symptom record is written. CICS appends to the `symrecs.nnnnnn` file and never truncates it.

CSMT is a transient data queue that contains messages about transactions. Messages written to CSMT include the date, time, region name, and principal facility, when there is one.

CSMT is in `/var/cics_regions/region/data` (on Open Systems) or `c:\var\cics_regions\region data` (on Windows NT).

The `console.nnnnnn` is located in `/var/cics_regions/regionName/data` (on Open Systems) or `c:\var\cics_regions\regionName\data` (on Windows NT).

Every time that a new CICS region starts, the number of the `console.nnnnnn` file is incremented by one.

Note: When a CICS region is cold-started, console.nnnnnn and CSMT are recreated. Any information previously stored in these files is lost. Make a copy of console.nnnnnn and CSMT before you restart a region after an error.

The NT Event log is also a valuable source of diagnostic information and should be reviewed using the Windows Administrative Tools, Event Viewer.

12.1.2 Encina trace messages

Encina trace messages can be located in the subdirectories `/var/cics_servers/` (on Open Systems) or `c:\var\cics_servers\` (on Windows NT).

These messages are generated by the server for fatal, nonfatal, and audit messages.

12.1.3 DCE diagnostic messages

DCE messages are written to `/opt/dcelocal/var/svc` (on Open Systems) or `c:\opt\cics\dcelocal\var\svc` (on Windows NT).

The files that store this data are `error.log`, `fatal.log` and `warning.log`.

DCE Endpoint mapper messages are written to `/opt/dcelocal/var/dced/dced.log`

DCE Security messages are written to `/opt/dcelocal/var/security/secd.log` or `/opt/dcelocal/var/security/adm/secd/secd.log`.

Database-related error data can be found in `DB2diag.log`

The message output is classified into five categories: fatal, error, warning, notice, and verbose notice. DCE itself is capable of logging messages to files that are stored by default in `/opt/dcelocal/var/svc`, and whose disposition is most easily controlled via the `/opt/dcelocal/var/svc/routing` file.

By default, only fatal, error and warning messages are actually kept; messages of type notice and verbose notice are discarded. The default messages are stored in files named `fatal.log`, `error.log`, and `warning.log` in the `/opt/dcelocal/var/svc` directory. You can cause DCE to retain notice and verbose notice messages too, by editing the `/opt/dcelocal/var/svc/routing` and adding lines like these:

```
NOTICE:FILE:/opt/dcelocal/var/svc/notice.log
NOTICE_VERBOSE:FILE:/opt/dcelocal/var/svc/verbose.log
```

The DCE endpoint mapper process, `dced`, keeps a log file in `/opt/dcelocal/var/dced/dced.log`. This log file is created anew each time `dced` is started; old messages from the previous instance of `dced` are not saved.

Similarly, on security server machines, `secd` keeps a log file in either `/opt/dcelocal/var/security/secd.log` or in `/opt/dcelocal/var/security/adm/secd/secd.log` (depending on the platform). This logfile is also created anew whenever `secd` is started, and old messages are not retained.

The `cicsservice` utility, found in the `cics/utls` directory, can be used to package up the diagnostic data for transmission, but, if you have failed to clean up your obsolete dumps and logs, this utility will package up a large amount of irrelevant data that has nothing to do with the current problem. This is probably best avoided if you do not manage your dump and log files.

12.1.4 WebSphere TXSeries tracing

Tracing in WebSphere TXSeries can require concurrent traces to be run for some or all of the key components, that is, CICS, Encina and DCE.

TXSeries CICS full trace using the Auxiliary Trace facility

Trace in WebSphere TXSeries 5.0 is written per thread per process, so if the `cicssas` process has 26 threads, then 26 files are recreated.

The suggested `CICSTRACE` environment variable setting is:

```
CICSTRACE= -A on -d /tmp -B off -M on -S on -t all=5
```

where:

- ▶ A = TraceFlagAux
- ▶ B = TraceFlag BUffer
- ▶ M = TraceMasterFlag
- ▶ S = TraceSystemFlag
- ▶ t = TraceSystemSpec.
- ▶ d = TraceDirectorySystem

Important: The blank before the `-A` *must* be present!

The `CICSTRACE` environment variable can be used to override the values from the `RD` stanza in the master trace area for any of the trace-related attributes.

This variable can be set either in the region environment file, `/var/cics_regions/regionName/environment` where it is read when the region starts, or it can be set on the command line, where it is read dynamically.

Starting TXSeries CICS system tracing

To start collecting system trace from a running region, issue the following commands:

```
CECI TRACE ON
CECI TRACE SYSTEM ON
CEMT SET AUXTRACE ON
```

Stopping TXSeries CICS system tracing

To stop collecting system trace from a running region, issue the following commands:

```
CEMT SET AUXTRACE OFF
CECI TRACE SYSTEM OFF
CECI TRACE OFF
```

TXSeries CICS trace files

The location of all system trace files, including those created by dynamic trace commands, is determined by the value of the `TraceDirectorySystem` attribute.

System trace records are stored directly to files, called auxiliary trace files, when the `TraceFlagAux` RD attribute is switched on. The location of all system trace files, including auxiliary trace files, is determined by the value of the `TraceDirectorySystem` RD attribute. Auxiliary trace files are named according to the following pattern:

`regionName.processName.nprocessNumber.pprocessID.tthreadID.format`

Under this naming convention, `processName` is the name of the CICS process from which the trace is generated; the `processNumber` is the identifier assigned by CICS, not the operating system; `processID` is the operating-system identifier. The `threadID` is omitted for single-threaded processes like standalone programs; in multi-threaded processes, the trace from each thread goes into a separate file. The `format` field indicates whether or not the file is formatted; the value is either `cicsfmt` for a formatted file or `cicstrc` for an unformatted file.

The Unformatted files need to be formatted by using the `cicstfmt` utility before they can be read.

12.2 Encina tracing for CICS Application Server processes

The following procedures will allow you to find the server name used by the `cicsas` so that `tkadmin` tracing can be turned on for a `cicsas` process. This is an

alternative to issuing the trace at startup in the `/var/cics_regions/<region>/environment` file.

This is a more dynamic form of tracing as we will start tracing a specific process for a specific server. It can be difficult to determine the server name which the cicsas process is using and which is required to set the correct trace value.

To find out the server name in relation to a specific CICS Application Server process, use the following procedure (`$=Unix prompt, as root, "\" = continuation`).

1. First, list the PIDs of the cicsas processes:

```
$ps -ef | grep cicsas
```

2. Now keep track of the PIDs listed; you will need them later.

3. We now need to determine the encina "server_id" of these processes, so that we can trace them. The best way to do this is to issue a log format command so that the cicsas server names are displayed. For example:

```
$cicslogfmt -r <region_name> -a
```

You will see a long listing of the cicsas server names. For example:

```
"ncadg_ip_udp:9.38.201.19[36441]" (found in the ASCII on the side).
```

You will see many names in the log format output, not all of them cicsas names. To verify that you have the right ones, select a likely server name like `CICS_AS_101_KMPREG1`, and query its PID, like this:

```
$tkadmin query process -server 'ncadg_ip_udp:9.38.201.19[36441]'
```

The output will be the PID of the cicsas process. Once you have matched all of the PIDs from step 2, you can start the traces.

4. To set the trace spec, do the following:

```
$tkadmin trace spec -server 'ncadg_ip_udp:9.38.201.19[36441]' tmtx=all
```

5. To verify that it is set correctly, do the following:

```
$tkadmin list trace -server 'ncadg_ip_udp:9.38.201.19[36441]'
```

You will see `tmtx` set to `trace_all` in the output.

6. Now redirect the trace to some file system:

```
$tkadmin redirect trace -s 'ncadg_ip_udp:9.38.201.19[36441]' trace -dest /tmp/somefile.tmp
```

We will now be sending all XA events to `/tmp/somefile.tmp`.

7. Once you have completed tracing, turn tracing off using the following procedure.

```
$tkadmin redirect trace -s 'ncadg_ip_udp:9.38.201.19[36441]' trace
$tkadmin trace specification -s 'ncadg_ip_udp:9.38.201.19[36441]'
all=default
```

The command will reset everything back to the default values.

12.2.1 Writing trace data to in-storage buffers

System trace records are stored in the trace buffer of each process when the TraceFlagBuffer is switched on. The trace buffer is a ring; trace records are written sequentially into the buffer, and when the end of the buffer is reached, storage continues at the beginning of the buffer, overwriting older records. The size of the ring buffer is determined by the value of the TraceMaxSizeBuffer attribute. The default value is 131 072 bytes. The attribute can be set to any positive integer, but changes do not affect processes already running.

The ring buffer is the default destination for system trace records. Storing trace records in the ring buffer puts the least load on the process being traced, making this the best way to trace production systems.

Records in the ring buffer must be retrieved before they can be read. This is achieved by dumping the ring buffer to a file. CICS processes dump their ring buffers automatically, depending on the following environmental variable settings.

- ▶ CICSTRACE_DUMP_ON_ABNORMAL_EXIT
- ▶ CICSTRACE_DUMP_ON_EXIT
- ▶ CICSTRACE_DUMP_ON_SYMREC
- ▶ CICSTRACE_DUMP_ON_ABEND
- ▶ CICSTRACE_DUMP_ON_MSN

The location of all system trace files, including those dumped from the buffer, is determined by the value of the TraceDirectorySystem environment variable.

For Encina, the **tkadmin dump ringbuffer** command dumps trace output contained in a main memory ring buffer to a file. The **tkadmin set ringbuffer size** command changes the size of a ring buffer.

12.3 CICS Transaction Gateway and CICS Universal Client

In this section, we will discuss where to locate the diagnostic message data and how to run tracing on the CICS Transaction Gateway and the CICS Universal Client for UNIX and Windows.

Messages

There are two types of message in the Client daemon. These are messages displayed to the user and errors written to the Client daemon error log and trace file. The *CICS Transaction Gateway: Gateway Messages* book explains all of these messages.

Any errors on the client workstation that are not caused by incorrect use of the API are written to the Client daemon error log. The error log (which has a default name of CICSCLI.LOG) is an ASCII text file that you can browse using a standard text editor.

Help information for all messages is provided in two ASCII text files in the <install_path>\bin subdirectory. You can view these using any standard text editor:

- ▶ CCLMSG.HLP - Help for the end user messages, in the language you chose at installation time.
- ▶ CCLLOG.HLP- Help for the trace and log messages. This is provided in English only.

Errors resulting from the incorrect use of the APIs simply result in error return codes to the application. It is the responsibility of the application to notify the end user of the error and provide guidance on corrective actions.

Pop-up messages

Errors generated from within the Client daemon are displayed in a pop-up window. You must click **OK** in the pop-up window before the Client daemon can continue processing. There may be times when you do not want messages to appear in pop-up windows, for example, if you leave the Client daemon running unattended overnight. To disable the display of pop-up messages, enter the following command:

```
CICSCLI -n
```

When the display of pop-up messages is disabled, the messages are still written to the Client daemon error log. To re-enable the display of pop-up messages, enter the following command:

```
CICSCLI -e
```

You can specify the `-n` parameter together with the `-s` parameter. The display of pop-up messages is enabled by default.

12.4 CICS Universal Client tracing

The Client daemon tracing is a very useful problem determination tool for communication problems. You can use the trace functions to collect detailed information on the execution of a certain function or transaction. A trace can show you how the execution of a particular activity is affected by, for example, the execution of other tasks in a CICS system. Each trace entry has a time stamp, which provides information on the time taken to perform certain activities.

You can specify which components of the Client daemon you want to trace. You control this with the **CICSCLI -m** command (see the CICSCLI command reference), or by specifying a list of components using the configuration tool.

The output from the trace function is a binary trace file called, by default, CICSCLI.BIN in the <install_path>\bin subdirectory. You can specify a different name for this file, using the configuration tool. However, you cannot change the .BIN extension. Using the Maximum Client wrap size configuration setting, you can specify that the binary trace file should wrap into a second trace file, and you can also specify the maximum size of these files.

To read the trace, run the **CICSFTRC** utility to convert the binary file or files into a text file. This text file is called CICSCLI.TRC by default. The default trace files are:

- ▶ CICSCLI.BIN - The binary trace file produced by running the Client daemon trace.
- ▶ CICSCLI.WRP - The second binary trace file if wrapping of client trace is enabled.
- ▶ CICSCLI.TRC - The name of the text trace file produced when the binary trace file is converted to a text file using the CICSFTRC utility.
- ▶ CICSCLI.BAK - The backup file of the binary trace file. A backup file is produced from any existing .BIN file when you turn tracing on.

See 12.5, “Formatting the binary trace file (CICSFTRC)” on page 311 for information on the trace conversion utility.

12.4.1 Starting and stopping Client daemon tracing

To start Client daemon tracing, enter the **CICSCLI** command with the **-d** option, for example: **CICSCLI -d=nnn** where nnn is optional, and is the maximum size in bytes of the data areas to be traced (the default value is 512).

It is recommended that you set at least **-d=1000** to ensure that all relevant data is included in the trace before sending it to your support organization.

As a general rule, the following is recommended for CICS Client tracing:

```
CICSCLI -d=9999 -m=all
```

If you need to trace the Client daemon immediately from startup, you can specify the `-s` and `-d` parameters together in the same **CICSCLI** command. For example, the following command starts the connection to a CICS server with the name **CICSTCP**, enables the trace function, and sets the maximum data area size to be traced to 128 bytes:

```
CICSCLI -s=CICSTCP -d=128
```

You can specify which components are to be traced when you start tracing. See the **CICSCLI** command reference for details.

To stop Client daemon tracing, enter the **CICSCLI** command with the `-o` option, for example:

```
CICSCLI -o
```

The trace is also automatically stopped if you stop the Client daemon by using the **CICSCLI -x** command.

Following are some definitions to help you choose which components to trace.

- ▶ **ALL** - This option traces everything. It is the preferred option; use it if performance allows, and consider using the binary formatting tool to filter information. See 12.5, “Formatting the binary trace file (CICSFTRC)” on page 311 for details.
- ▶ **API.1** and **API.2** - These trace the boundary between the user application or Java classes, and the Client daemon. Switching on **API.2** automatically switches on **API.1** as well.
- ▶ **DRV.1** and **DRV.2** - The protocol drivers trace the boundary between the Client daemon and the network protocol. Specify the **DRV.1** and **API** components if you are not sure whether a problem is inside the Client daemon, and you want to minimize the impact on performance, for example when trying to determine a performance problem.

You can also specify these components to help determine which product is causing the system to lock up.

- ▶ **CCL** - This component traces the main Client daemon process. Specify the **API** and **CCL** components if you believe that the problem is within the Client daemon.
- ▶ **TRN** - The **TRN** component traces the internal interprocess transport between Client processes. Use it if entries in the Client log refer to functions such as **FaarqGetMsg**, **FaarqPutMsg**, or **FaarqStart**. **TRN** is the most verbose tracing component.

12.4.2 Wrapping the Client daemon trace

You can control the size of the binary trace file by specifying that it wraps into a second trace file. You turn on wrapping of trace using the Maximum Client wrap size configuration setting, where you specify the maximum size of the wrapping trace (in kilobytes). If this value is 0 (the default), wrapping trace is not turned on.

When wrapping trace is turned on, two files (called CICSCLI.BIN and CICSCLI.WRP) are used. Each file can be up to half the size of the Maximum Client wrap size value.

12.5 Formatting the binary trace file (CICSFTRC)

You use the Binary Trace Formatter utility **CICSFTRC** to convert the binary trace file CICSCLI.TRC to ASCII text. The utility has the following parameters:

- ▶ **-m=<list_of_components>**
Specifies that only trace points from the listed components are written to the text file. The components you can specify are the same as for **CICSCLI -m**. If **-m** is not specified, all trace points in the binary trace are written to the text file.
- ▶ **-w[=filename]**
Indicates that there are two binary trace files to format and then concatenate (that is, the binary files were created with a wrapping trace). If no filename is specified with the **-w** parameter, **CICSFTRC** assumes that the name of the second trace file is **CICSCLI.WRP**.
- ▶ **-n**
Indents entry and exit points in the test trace file to make it more readable. By default, indentation is turned off.
- ▶ **-d**
Specifies detailed trace formatting. If you are using EPI calls, **CICSTERM** or **CICSPRINT**, an approximation of the screen layout will be included in the trace.
- ▶ **-i=filename**
Specifies the name of the input (binary) trace file, which is **CICSCLI.BIN** by default.
- ▶ **-o=filename**
Specifies the name of the output (text) trace file. If no **-o** parameter is specified, the name of the text trace file is assumed to be **CICSCLI.TRC**.

- ▶ **-f**
Overwrite any existing files.
- ▶ **-s**
Summary trace formatting is driven from a text file (CCLSUMTR.TXT) which is read in at initialization time. This defines the set of trace points for which you want summary tracing, and the type of trace point. As DetailFormat reaches each trace point, if it is one of those read in from this file, a line is generated in the summary file. Use as requested by your IBM support organization; see the program support.

12.6 Client daemon trace analysis

The Client daemon trace file records detailed information on all actions taken during the execution of a particular activity. You can use this information in your problem determination activities, and to help understand how the Client daemon performs a particular function, for example, establishing a connection to a CICS server.

Format of trace entries

The entries in the Client daemon trace file have the following format:

```
time [process id,thread id] [number] component trace_message data
```

where:

- ▶ **time** - The time the entry was written, to millisecond accuracy.
- ▶ **[process id, thread id]** - Process id is a unique number that the operating system uses to identify a process. Thread id is a unique number that the operating system uses to identify a thread within a particular process.
- ▶ **[number]** - A number to aid your support organization in the diagnosis of serious problems.
- ▶ **[component]** - The component of the product to which this entry applies.
- ▶ **trace message** - The trace message number and text. These trace messages are explained in *CICS Transaction Gateway: Gateway Messages*.
- ▶ **data** - Some trace entries include a dump of key data blocks in addition to the trace message.

12.7 CICS Transaction Gateway tracing

To start full tracing on the CTG at startup, you issue the following command:

```
ctgstart -x -tfile=ctg.trc
```

This will create a trace file called `ctg.trc` as indicated by the `-tfile` parameter. The `-x` option indicates full tracing.

JNI tracing is not enabled through the normal Gateway tracing options. To enable it, you need to set an environment variable `CTG_JNI_TRACE` and specify the path and file where you want the information stored. For example:

```
CTG_JNI_TRACE=c:\temp\ctgjni.trc
```

To turn on full tracing for the Java components of CTG, the following needs to be added to the application code. All output will go to standard out so you will need to pipe it to a file. Add the import to the MAIN of the program. This will load the debug methods.

```
import com.ibm.ctg.client.T
```

Also, within the `main()` method, insert the following lines:

```
// user code begin {__main()_1}  
T.setDebugOn(true);  
// user code end {__main()_1}
```

This will set the debug trace on. To set it off, use the same block of code substituting `(false)` for `(true)`.

To do exception only tracing, use the same code as above but modify it:

```
T.setStackOn(true);
```

Additional data that is required includes the `CICSCLI.ERR` file, the `CICSCLI.LOG` file and also the `CTG.ini` file.

12.8 CICS Transaction Gateway on z/OS

The Gateway daemon is used only in remote mode; it listens on protocols defined in `CTG.INI` for gateway requests from remote Java client applications. It issues these requests to the Client daemon on distributed platforms, and directly to CICS over the EXCI on z/OS. The Gateway daemon runs the protocol listener threads, the worker threads and the connection manager threads. It uses the `GATEWAY` section of `CTG.INI` (and on z/OS the `ctgenvvar` script) for its configuration.

The Client daemon process, `cclclnt`, exists only on distributed platforms. It manages network connections to CICS servers. It processes ECI, EPI, and ESI requests, sending and receiving the appropriate flows from the CICS server to satisfy the application requests. It uses the CLIENT section of CTG.INI for its configuration.

The Gateway classes are the interface for Java Client applications to connect to the Gateway daemon. The Gateway classes, which are supplied with the CICS Transaction Gateway, must be in the classpath for Java Client applications to run.

The Java Client application is a Java application, servlet or applet that communicates with the Gateway classes.

12.8.1 The Gateway daemon

The Gateway daemon consists of two parts.

1. The first is the listeners component that accepts connections (TCP, HTTP) from the remote Java client application and Gateway classes. In the CICS TG file system, this is file `ctgserver.jar` in the classes subdirectory and it is written in Java.
2. The second component makes the connection to the CICS Server using the External CICS Interface (EXCI). This component is the `libCTGJNI.so` file in the bin subdirectory and is written in C.

The two components communicate using the Java Native Interface (JNI) which allows Java code to interact with components written in C.

The Gateway daemon listener accepts the connections from the Gateway Classes and passes them through to the JNI component of the Gateway daemon which makes EXCI connections into the CICS server.

12.8.2 Trace file allocation

You control the size of trace files by specifying the `-tfilesize=<size>` option in conjunction with `-tfile=<filename>` on the `ctgstart` command. The `<size>` parameter is a value in kilobytes that should be greater than 4 and indicates the maximum size of the trace file. This will create a wrap-around file that will never exceed the maximum file size specified. A review of the timestamps is required to check for newest and oldest entries in the trace. The top of the trace file will not necessarily be the start, or oldest entries.

You can also limit the size of the hex dumps and obtain certain sections of the hex dump. Specify `-truncationsize=<size>` where `size` is the maximum size of

the hex dump you want, measured in bytes (not kilobytes as for `tfilesize`). Use `-dumpoffset` to start the hex dumps at a certain offset into the byte array, for example:

```
ctgstart -x -tfile=myctg.trc -tfilesize=100 -truncationsize=100
-dumpoffset=15
```

This will create a file called `myctg.trc` that contains a wrap-around gateway trace with a maximum file size of 100 KB. Every hex dump will be displayed starting at the 15th byte and will show only the first 100 bytes of the dump.

When using the `tfilesize` limit, it is crucial to have the timestamp entries or the file will be impossible to interpret. However, if you have not set the `tfilesize` limit then you may wish to trace without timestamps. This can be achieved by passing a `-notime` parameter to **ctgstart**:

```
ctgstart -x -tfile=gway.trc -notime
```

This last one is not recommended; the timestamp is crucial when attempting to match client and server events. Not having timestamps makes that almost impossible.

12.8.3 Application trace

Application trace will typically be used where problems are arising with some component of the Java client application or its interaction with the Gateway daemon. It can be used to determine the requests and data areas the client application is creating. This is useful to analyze client application and CICS server interaction. The timestamps are useful since they can be used to measure accurately the overall time required to interact with CICS. This helps to isolate the CICS interaction component of any performance measurements.

Performance problems can be investigated from here first to check that there is no problem with the sending of data, for example because of incorrect `COMMAREA` sizes.

The first way to enable Application trace is to add additional code to a Java client application to interact with the Tracing API. This API is provided through the `T` class, part of the Gateway classes component of the CICS TG.

The `T` class allows the application itself to activate and deactivate trace at any time and to control the levels of trace.

Java application programmers can control the tracing from their application by importing the `com.ibm.ctg.client.T` class and then make calls to the static methods on the `T` class, for example:

```
import com.ibm.ctg.client.T;
T.setDebugOn(true);
```

This will activate full tracing.

Alternatively, you can enable Application tracing by using Java system directives. Using system directives is a quick and easy way of activating application trace. For example, it can be used to quickly see additional information about the attempted interactions with the Gateway daemon.

The disadvantage is that you do not have as much control of what and when to trace as you have by using the tracing API (T class) in an application. For example, in a WebSphere Application Server environment, you must specify the `-D` option on the server's Java Virtual Machine (JVM). However, this JVM may be shared by multiple threads of your application and perhaps by additional applications. If the system property has been activated then all of the threads and applications will be affected and so will all write trace output at the same time. If this is a problem, consider using the API control to only activate trace for the required parts of an application server environment.

The following shows the system directives and their equivalent API calls:

```
gateway.T = on/off => setDebugOn()
gateway.T.stack = on/off => setStackOn()
gateway.T.trace = on/off => setTraceOn()
gateway.T.timing = on/off => setTimingOn()
gateway.T.fullDataDump = on/off => setfullDataDump()
gateway.T.setTruncationSize = integer => setTruncationSize()
gateway.T.setDumpOffset = integer => setDumpOffset()
gateway.T.setTFile = String => setTFile(true,String)
gateway.T.setJNITFile = String => setJNITFile(true,String)
```

12.8.4 Gateway daemon trace

The Gateway daemon trace is useful for identifying exactly which requests have reached the gateway. The trace outputs all of the request parameters and can be used to ensure that the client applications are successfully passing their requests to the Gateway daemon with the correct parameters.

The Gateway daemon trace can be started automatically at CICS start using the appropriate **ctgstart** option or dynamically via TCPAdmin.

TCPAdmin is a new protocol handler that is similar in configuration and operation to the standard TCP protocol handler. It is also configured through the CTG.INI configuration file and uses the same TCP protocol as its basis. There is a GUI front-end that can be used to dynamically start tracing as well as change the CTG.INI file.

The **ctgstart -stack** option will turn on exception stack tracing only and most Java exceptions are traced including expected exceptions during CTG normal operation. No other tracing is performed.

ctgstart -trace enables standard tracing. The trace includes the first 80 bytes of the COMMAREA by default.

ctgstart -x enables full debug tracing, which includes everything traced by the **-trace** option with additional internal information. This includes the entire COMMAREA by default. This can produce large amounts of data and the HFS trace file size must be considered as well as the significant negative impact on performance.

Specifying the **ctgstart** trace options causes the Gateway daemon to make calls to the T class at startup, in the same way that a Java client application does to control application tracing.

12.8.5 JNI tracing

JNI trace is useful in determining what caused an ECI_ERR_XXX condition. The JNI trace captures the interaction of the Gateway daemon and the EXCI (External CICS Interface) to enable CTG and host CICS communication/interaction problem diagnosis. The JNI trace can also be used to assist with RACF® authentication problems where the return from RACF is shown in the trace; this can be used to determine reasons for security failures. The JNI trace will also be helpful in identifying system-related configuration problems on CTG for z/OS. It is less suited to application-related errors where Application trace or Gateway daemon trace can more effectively detail information about each request.

JNI tracing is enabled for the lifetime of the Gateway daemon by specifying a Java directive of `gateway.T.setJNITFile=<filename>`. This is done by using the **-j** option on the **ctgstart** command to pass a parameter to the JVM. Here is an example:

```
ctgstart -j -Dgateway.T.setJNITFile=jni.trc
```

You can also start the JNI trace from within the Java client application's JVM. This means that you can switch on JNI tracing from within your Java client application by using the T class static method, `setJNITFile(boolean, String)`. You can also pass a Java directive to the Java client application's JVM to switch on JNI tracing at startup, without writing any application code; for example:

```
java -Dgateway.T.setJNITFile=jni.trc com.my.Application
```

Finally, if you are using a Gateway daemon (remote mode), you can use the TCPAdmin function to allow dynamic activation and deactivation of JNI trace. JNI

logs all error interactions with the EXCI automatically and these are written to \$HOME/ibm/ctg/ctgjinilog.<process id>, where \$HOME is an environment variable, typically /u/<ctguser> and <ctguser> is the user ID under which the Gateway daemon runs. If \$HOME does not exist then the user's current directory is used. <process id> is the process ID of the Gateway daemon.

You can issue the following command under USS to determine the location of the files:

```
ps -ef | grep JGate
```

The log may already have sufficient data to assist with problem determination, thus avoiding the need to use JNI trace.

12.8.6 EXCI trace

The EXCI trace shows activity on the interface between CTG and CICS Server and is stored in the Gateway region address space, or GTF.

To use GTF for EXCI tracing, GTF user tracing must be active. GTF must be started in the z/OS image where the CTG address space you are going to trace resides, and you must specify GTF=ON in the DFHXCOPT options table. If you use GTF trace for both the CICS server region and the EXCI region, the trace entries are interleaved, which can help you with problem determination in the CICS-EXCI environment. The external CICS interface does not support any form of auxiliary trace.

To enable GTF tracing, you must set the DFHXCOPT module in the following format:

```
DFHXCO TYPE={CSECT|DSECT}
[,CICSSVC={0|number}]
[,CONFDATA={SHOWHIDETC}]
[,DURETRY={30|number-of-seconds}]
[,GTF={OFF|ON}]
[,MSGCASE={MIXED|UPPER}]
[,SURROGCHK={YESNO}]
[,TIMEOUT={0|number}]
[,TRACE={OFF|1|2}]
[,TRACESZE={16|number-of-kilobytes}]
[,TRAP={OFF|ON}]
END DFHXCOPT
```

You must specify the EXCI options table in **ctgenvvar** EXCI_OPTIONS='YOUR.EXCI.LOADLIB', where 'YOUR.EXCI.LOADLIB' contains the assembled DFHXCOPT.

Specify the trace level you require in DFHXCOPT and then assemble the table into a dataset such as YOUR.EXCI.LOADLIB.

ctgenvar builds the STEPLIB environment variable automatically from any components that must be on the STEPLIB.

By specifying EXCI_OPTIONS='YOUR.EXCI.LOADLIB' you are ensuring that your assembled DFHXCOPT table will be added to the CICS TG's STEPLIB value because **ctgenvar** contains the following line:

```
export STEPLIB=${STEPLIB}:${EXCI_OPTIONS}:${EXCI_LOADLIB}
```

If you decide to specify environment variables in an alternative way, make sure that STEPLIB contains your EXCI options dataset.

Also ensure that the CICS job has your assembled DFHXCOPT table in the STEPLIB. If you do not do this then you may find that the CICS Transaction Gateway trace points (numbered 800x) will not be seen in the trace.

12.9 CICS problem diagnosis

The first reference points for all CICS problems are:

- ▶ System log (SYSLOG)
- ▶ CICS Job log
- ▶ System error data (SYS1.LOGREC)
- ▶ Message and codes manuals

The logs can be viewed using the z/OS Spool Display and Search Facility (SDSF).

The next sections provide you with some fundamental CICS problem determination techniques.

12.9.1 CICS messages

CICS messages consist of the prefix DFH followed by a two-letter component identifier (cc), and a four-digit message number (nnnn). The component identifier shows the domain or the component which issues the message. Some examples of the component identifier are:

- ▶ AP - The application domain
- ▶ DS - The dispatcher domain
- ▶ SM - The storage manager domain

- ▶ XM - The transaction manager
- ▶ XS - The CICS security component

Thus, the CICS message DFHAP0002 is issued from the application domain, identified by the two-character identifier AP.

12.9.2 CICS internal trace

Possibly the most useful diagnostic information that can be reviewed in a CICS SVC dump is the CICS internal trace. A record of all activity within the CICS region is stored. The default tracing options only capture exception trace entries. Unfortunately, we usually like to review what preceded the generation of an exception condition, since this is the only way to see what was the cause, not just the result.

Another unfortunate default is the CICS internal trace size which is set to 64K. In a busy CICS region, this would store less than one second of trace data, hardly enough to enable you to review the flow of the transaction that caused the exception condition. We recommend an internal trace size of at least 2500K. In fact, 5000K is probably an optimum size. This should provide you with sufficient trace information, except in exceptionally busy systems. In a large, high usage environment, a 10000K trace table can hold as little as five seconds of trace data. While most CICS transactions are of short duration, it is good to be able to review a trace that includes the start of the transaction.

CICS tracing is performed by the trace domain at predetermined trace points in the CICS code during the regular flow of control. This includes user tracing from applications. Tracing is performed when you turn on CICS internal tracing, auxiliary tracing, and GTF tracing. You can control the type of tracing to suit your needs, except when an exception condition is detected by CICS. CICS always makes an exception trace entry. You cannot turn exception tracing off. Trace points are included at specific points in CICS code; from these points, trace entries can be written to any currently selected trace destination. All CICS trace points are listed in alphanumeric sequence in the *CICS User's Handbook*, SX33-1188.

Level-1 trace points are designed to give you enough diagnostic information to fix "user" errors. Level-2 trace points are situated between the level-1 trace points, and they provide information that is likely to be more useful for fixing errors within CICS code. You probably will not want to use level-2 trace points yourself, unless you are asked to do so by IBM support staff after you have referred a problem to them.

It is recommended that you trace all CICS components at level 1 and that CICS tracing be active all the time. Lack of sufficient trace data can delay problem resolution, and exception only trace data is not sufficient in most cases.

12.9.3 CICS Trace Control Facility (CETR)

CICS exception tracing is always done by CICS when it detects an exception condition. The sorts of exception that might be detected include bad parameters on a domain call, and any abnormal response from a called routine. The aim is "first failure data capture," to record data that might be relevant to the exception as soon as possible after it has been detected. The trace options can be set using the CICS/ESA® Trace Control Facility (*CETR*) transaction. This enables you to increase the trace table size and lets you control CICS component (domain) trace options. Tracing for all CICS components should be set to level 1 when collecting diagnostic data.

You can also write out CICS trace data to a disk dataset using the AUXTRACE facility. This is also controlled via the CETR transaction and can have the benefit of not requiring the CICS region to be dumped to review the data. This is a good option for tracing a recreatable scenario. You can also request trace data to be collected for a specific transaction, or terminal.

CICS auxiliary trace entries are directed to one of two auxiliary trace data sets, DFHAUXT and DFHBUXT. These are CICS-owned BSAM data sets, and they must be created before CICS is started. They cannot be redefined dynamically. The amount of data that can be collected is related to the size of the auxiliary trace data sets. You can use the AUXTR system initialization parameter to turn CICS auxiliary trace on or off in the system initialization table. Alternatively, You can select a status of STARTED, STOPPED, or PAUSED for CICS auxiliary trace dynamically using the CETR transaction.

The AUXTRACE data can be formatted using the supplied DFHTRnnn program, where nnn represents the CICS release, for example, 530 for TS 1.3 and 620 for TS 2.2.



Part 3

Appendixes



Problem determination tools: other platforms

This appendix collects problem determination tools for platforms other than AIX and WebSphere. During this redbook project, we discovered and documented many tools and instead of leaving them out, we have included them in this appendix.

Obviously, this appendix cannot cover all the other tools of other platforms; for more information about tools that you could not find here, refer to the product documentation.

Windows 2000 Server

Version: Windows 2000 Server, Service Pack 4

On Windows 2000, there are different tools that can aid in problem determination. You can review system performance and logs to investigate system related problems or product related problems that are hard to determine from product logs.

Task Manager

To review running processes and determine system CPU and memory usage, use Task Manager.

To open Task Manager, right-click the taskbar and select **Task Manager**. Or press **Ctrl-Alt-Delete** and select **Task Manager**.

To review running processes, select the **Processes** tab. To find Java processes that represent WebSphere Application Server processes, click the **Image Name** column to resort the list and search for `java.exe`. To end a process, select the process name and click **End Process**. For PIDs to WebSphere Application Server processes, see the Tools section of WebSphere Application Server for the `<server-name>.pid` file.

To monitor CPU and memory usage, switch to the Performance tab.

Event Viewer

Event Viewer displays application, security and system logs for your Windows system. You may need to review these logs to look for errors messages or warnings. You may also need to look for WebSphere MQ errors messages in the application log.

To open Event Viewer, go to **Start -> Control Panel -> Administrative Tools -> Event Viewer**. Then switch between Application Log, Security Log, and System Log.

In the Application Log, you can find WebSphere MQ messages and errors. Double-click a message to read the description of the problem.

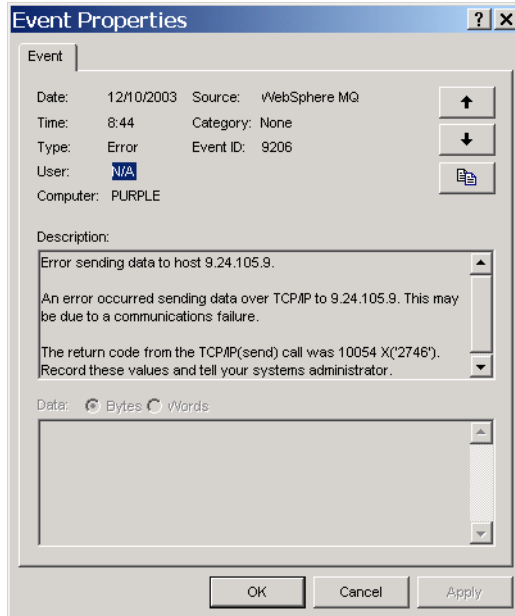


Figure A-1 Event Viewer with WebSphere MQ error message

Performance

Using the Performance tool in Windows allows you to pull together many resources and monitor them. You can monitor TCP connections and processor time or add WebSphere MQ and monitor queue depth. You can monitor your local machine or a remote machine.

To open the Performance tool, go to **Start -> Control Panel -> Administrative Tools -> Performance**.

To add WebSphere MQ, select **Console -> Add/Remove Snap-in**. Then click **Add** and select **WebSphere MQ**. Generally, to select a snap-in for a remote machine, you can double-click the snap-in.

To add items to the System Monitor view in order to monitor their progress, click the **+** (plus) button. Select the system to monitor, the type of performance object and counters. To monitor queue depth, add WebShere MQ as a snap-in, then select **MQSeries Queues** for the performance object and **Queue depth** as a counter.

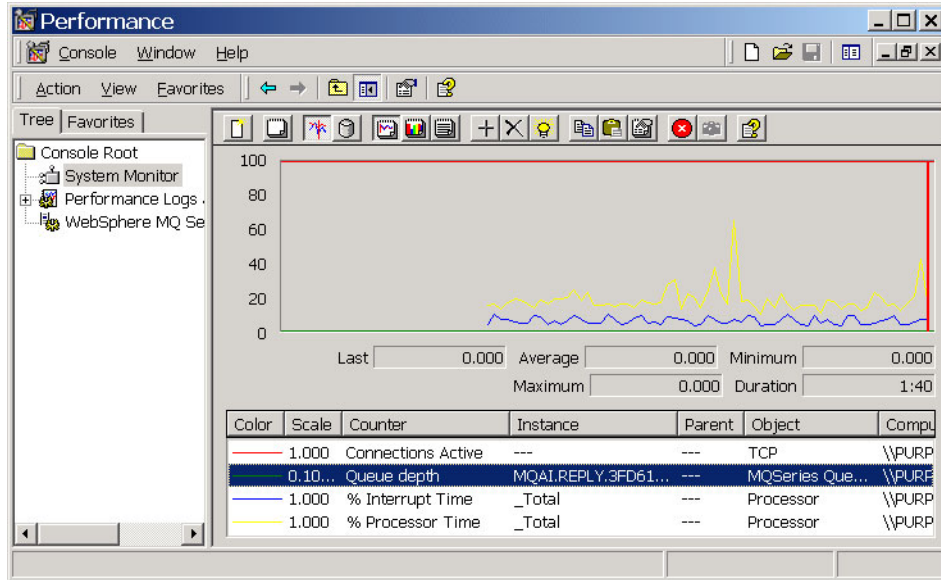


Figure A-2 Monitoring various activities in Performance monitor

Registry Editor

Reviewing the Registry Editor can help you confirm what products you have installed or what your system thinks is installed. Use the registry with extreme caution. If you make a mistake, you can disrupt your system. Start the Registry Editor with regedit.exe. Look for product keys, for example:

HKEY_LOCAL_MACHINE\SOFTWARE\IBM\WebSphere Application Server or
HKEY_LOCAL_MACHINE\SOFTWARE\IBM\MQSeries.

Tip: Restarting a Windows system, while not necessarily a problem determination tool, can sometimes be the solution to strange problems.

Java

Besides the javacore dump, on Windows platforms, the DrWatson debugger can provide additional information. Make sure you save the information from DrWatson when you get an error and submit that information to the support team.

WebSphere MQ: Windows

Assuming that WebSphere MQ has been installed on the C: drive in the MQM directory, the following information will help you locate the required logs.

If the queue manager name is known and the queue manager is available, error logs are located in `c:\mqm\qmgrs\qmname\errors`.

If the queue manager is not available, error logs are located in:
`c:\mqm\qmgrs\@SYSTEM\errors`.

If an error has occurred with a client application, error logs are located on the client's root drive in `c:\mqm\errors`.

In WebSphere MQ for Windows NT/2000, an indication of the error is also added to the Application Log, which can be examined with the Event Viewer application.

You can also examine the Registry to help resolve any errors. The Registry Editor, supplied with Windows, allows you to filter errors that are placed in the Event Log by placing the code in the following Registry entry:

```
HKEY_LOCAL_MACHINE\SOFTWARE\IBM\MQSeries\CurrentVersion\IgnoredErrorCodes
```

For example, to ignore error 5000, add `AMQ5000` to the list.

Tracing

To start WebSphere MQ tracing, issue the `strmqtrc` command.

In Windows, the trace files are located in the `c:\mqm\errors` directory (assuming drive C: has been used).

To end the WebSphere MQ trace, issue the `endmqtrc` command.

To format the WebSphere MQ trace file, issue the `dspmqtrc` command. You must specify the input (unformatted trace file, for example: `AMQxxxxx.TRC`) and the output file name.

Error logs

Assuming that WebSphere MQ has been installed on the C: drive in the MQM directory, the following statements are true.

If the queue manager name is known and the queue manager is available, error logs are located in `C:\mqm\qmgrs\QM_NAME\errors`.

If the queue manager is not available, error logs are located in
`c:\mqm\qmgrs\@SYSTEM\errors`.

If an error has occurred with a client application, error logs are located on the clients root drive in `c:\mqm\errors`.

In WebSphere MQ for Windows NT/2000 only, an indication of the error is also added to the Application Log, which can be examined with the Event Viewer

application provided with Windows NT/2000. You can also examine the Registry to help resolve any errors. The Registry Editor supplied with Windows NT/2000 allows you to filter errors that are placed in the Event Log by placing the code in the following Registry entry:

```
HKEY_LOCAL_MACHINE->SOFTWARE->IBM->WebSphereMQ->Current Version->Ignored  
Error Codes
```

WebSphere Business Integration Message Broker: Windows

This section provides problem determination information for WebSphere Business Integration Message Broker to perform tracing for:

- ▶ ODBC trace
- ▶ WebSphere Business Integration Configmgr and Broker trace

ODBC trace

Initiating a trace is platform-dependent. On Windows, use the Tracing tab of the ODBC function (see Figure A-3 on page 331), as follows:

1. Click **Start -> Settings -> Control Panel -> Administrative Tools**.
2. Double-click **Data Sources**.
3. Select the **Tracing** tab.
4. Click the **Start Tracing Now** button.
5. Click **OK**.

To stop ODBC tracing, go to the same panel and click the **Stop Tracing Now** button, then click **OK**. The file in which the trace is written is shown below the button and can be read using any editor; no formatting is required.

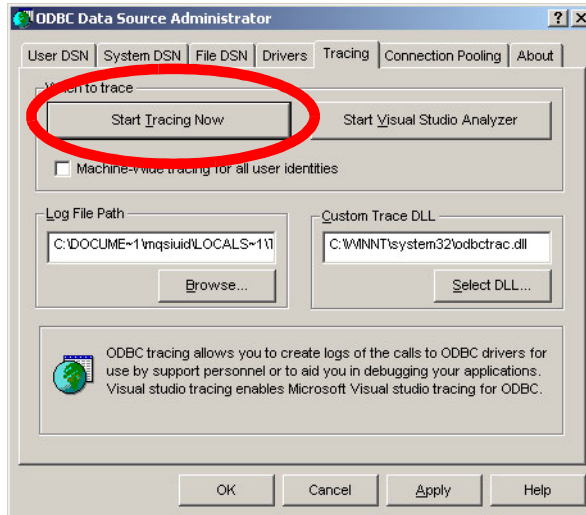


Figure A-3 The trace tab of the ODBC Data Source Administrator for Windows 2000, with Start Tracing Now button circled

WebSphere Business Integration Configmgr and Broker trace

1. Stop the Configuration Manager.
2. Run **regedit**.
3. Go to **HKEY_LOCAL_MACHINE/SOFTWARE/IBM/MQSeriesIntegrator/2/configmgr**. Right-click the **Broker**, add a new String Value, rename it to Trace and modify the Data value to debug.
4. Start the Configuration Manager by running **mqsi start**.
5. Recreate the problem.
6. Make sure the Configuration Manager is stopped.
7. Run **mqsi readlog configmgr -t -b agent -f -o agent.xml**.
8. Run **mqsi readlog configmgr -t -b service -f -o service.xml**.
9. Use **mqsi formatlog** to format the generated traces and send them in.
10. Return to the registry and delete the Trace entry created earlier.

z/OS

This section provides information on z/OS problem determination.

System Management User Interface debug utilities

The Administration and Operations applications of the System Management User Interface (SMUI) include a message log, a trace writer, a communication trace, and a debug facility that are intended to be used under the supervision of IBM support personnel. These facilities can be triggered by login options and menu bar actions.

The SMUI has different options for debugging. The following options can be selected using the SMUI Graphic User Interface (GUI).

▶ Message log

In the WebSphere for z/OS Administration application, select **File -> Message log...** (or press the keys **Ctrl-M**). A new window with the message log will appear.

▶ Trace writer

There are two ways to activate the trace writer:

- When you start the WebSphere for z/OS and OS/390 Administration, in the login window in **Options...**, select **Trace writer** and click **Set**. Now log in; the trace writer is started from the beginning.
- With WebSphere for z/OS and OS/390 Administration started, select **Options -> Diagnose -> Trace writer**. This message will appear at the bottom of the window: BB0N0701I Trace writer started. The trace writer is now active.

To see the trace writer log, select **Options -> Diagnose -> Trace viewer**.

You can save the trace writer in a file by selecting **Options -> Diagnose -> Trace file**.

▶ Communications trace

You can only start the communication trace in the login window; go to **Options...**, then select **Communications Trace** and click **Set**. Log in and the communication trace is started from the beginning.

To see or save the trace to a file, go to **Options -> Diagnose -> Trace viewer** or **Trace file**.

Use the following steps to turn on communications trace when you cannot connect to the host with the SM EUI.

- a. Open a DOS command prompt window and switch to the WebSphere installation directory on your workstation (for example, c:\Program Files\IBM\WebSphere for z/OS).
- b. Switch to the bin directory by issuing **cd bin**.
- c. Invoke the SM EUI application by issuing the following command:
`bbonrun >trace.1st 2>trace2.1st`
- d. When the login panel is displayed, select **Options....**
- e. Ensure the **Communications** trace option is selected.
- f. Click **Set**.
- g. Log in normally.

The trace data is not written to the files until the SM EUI is closed. The most useful data will appear in the trace2.lst file.

► **Debug mode**

You can activate this mode in two ways:

- In the login window in **Options...**, click **Debug mode -> Set**. Now you can log in with the debug mode started.
- You can also select **Options -> Diagnose -> Debug mode** when the WebSphere for z/OS and OS/390 Administrator is started.

To see the trace with the debug mode activated, select **Options -> Diagnose -> Trace viewer**.

You can save this trace in a file by selecting **Options -> Diagnose -> Trace file**.

Type of information received

The information that you can get from these four kinds of traces, as explained in the following sections, will be helpful to the IBM Support Center when diagnosing problems at a detailed level.

Message log

The message log contains a list of messages issued by the application. There is one message log for the Administration application, and another message log for the Operations application.

The messages in the message log are in chronological order. New messages are added to the bottom of the log. You can scroll the message log up and down to see older or newer messages.

Trace writer

The trace writer is used for support purposes only. It will impact performance and is to be used under the direction of IBM support personnel. The trace writer action controls the trace writer, which causes the trace entries to be written to the internal trace table and/or the output (file or viewer). You can start the trace writer and specify a trace level that determines which trace entries are written to the output.

Communications trace

The communications trace option starts the tracing of communication between the application and the Systems Management Server on z/OS or OS/390. The trace output is the IIOB communication packets being sent back and forth between the SMUI client and the WebSphere servers.

The SMUI is a CORBA application, so it uses the naming server to access the name space. J2EE applications use the LDAP server. During the WebSphere bootstrap process, an indirect reference to the naming server is stored in the System Management database to speed up the look-up at connect time. At this point, the request is received by the naming server (BBONM) and queued to WLM. The server region (BBONMS) starts. If you do not see that, the failure occurred earlier, most likely in step 3.

Debug mode

The Debug mode action is for support purposes only and is to be used under the direction of IBM support personnel. It enables various internal diagnostic facilities that may impact performance.

Language Environment® (CEEDUMP)

When an error occurs within the Language Environment (LE) or Java environment, often a CEEDUMP is written. It can be specified via DD Card or, if in a dump, via the IPCS command `IP VERBX CEEERRIP 'CEEDUMP'`. Normally the CEEDUMP data is written to job logs.

CEEDUMP data can help you to find the failing module by referencing the trace back data.

WebSphere MQ: z/OS

WebSphere MQ for z/OS provides unique messages which, together with the output of dumps, are aimed at providing sufficient data to allow diagnosis of the problem without having to try to reproduce it.

WebSphere MQ for z/OS tries to produce an error message when a problem is detected. All diagnostic messages begin with the prefix CSQ. Each error message generated by WebSphere MQ is unique; it is generated for one and only one error. Information about the error can be found in *WebSphere MQ for z/OS Messages and Codes*, GC33-0819. The first three characters of the names of WebSphere MQ for z/OS modules are also CSQ. The fourth character uniquely identifies the component. Characters five through eight are unique within the group identified by the first four characters. There may be some instances when no message is produced, or, if one is produced, it cannot be communicated. In these circumstances, you might have to analyze a dump to isolate the error to a particular module.

When capturing diagnostic data to analyze MQ problems, ensure that you dump both the MQ main (MSTR) and the channel initiator (CHIN) address spaces as well as the CHIN dataspace, CSQXTRDS. The following example shows the procedure to dump the MQ address spaces.

```
DUMP COMM=(MQSERIES MAIN DUMP)
*01 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
R 01,JOBNAME=(CSQ1MSTR,CSQ1CHIN),CONT
*02 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
IEE600I REPLY TO 01 IS;JOBNAME=(CSQ1MSTR,CSQ1CHIN),CONT
R 02,DSPNAME=('ssidCHIN'.CSQXTRDS),CONT
IEE600I REPLY TO 02 IS;DSPNAME=('ssidCHIN,CSQXTRDS),CONT
R 03,SDATA=(CSA,RGN,PSA,SQA,LSQA,TRT,SUM),END
IEE600I REPLY TO 03 IS;SDATA=(CSA,RGN,PSA,SQA,LSQA,TRT,SUM),END
IEA794I SVC DUMP HAS CAPTURED: 869
DUMPID=001 REQUESTED BY JOB (*MASTER*)
```

You can obtain information about API calls and user parameters passed by some WebSphere MQ calls upon entry to, and exit from, WebSphere MQ. To do this, you should use the global trace in conjunction with the MVS™ generalized trace facility (GTF). To use the trace for problem determination, you must start the following:

- ▶ The GTF for your MVS system. An example of the GTF start process follows:

```
START GTF.procname,,,(MODE=INT)
HASP100 GTF.procname ON STCINRDR
HASP373 GTF.procname STARTED
*01 AHL100A SPECIFY TRACE OPTIONS
R 01,TRACE=JOBNAMEP,USRP
TRACE=JOBNAMEP,USRP
IEE600I REPLY TO 12 IS;TRACE=JOBNAMEP,USRP
*02 ALH101A SPECIFY TRACE EVENT KEYWORDS - JOBNAME=,USR=
R 02,JOBNAME=(ssidMSTR,jobname),USR=(5E9,5EA,5EB,5EE,FC6)
JOBNAME=(ssidMSTR,jobname),USR=(5E9,5EA,5EB,5EE,FC6)
IEE600I REPLY TO 13 IS;JOBNAME=(ssidMSTR,jobname),USR=(5E9,5EA,5EB,5EE,FC6)
*03 ALH102A CONTINUE TRACE DEFINITION OR REPLY END
```

```

R 03,END
END
IEE600I REPLY TO 14 IS;END
AHL103I TRACE OPTIONS SELECTED-USR=(5EA,5E9,5EB,5EE,FC6)
AHL103I JOBNAME=(ssidMSTR,jobname)
*04 AHL125A RESPECIFY TRACE OPTIONS OR REPLY U
R 04,U
U

```

- ▶ The WebSphere MQ trace for each queue manager subsystem for which you want to collect data.

When you start the GTF, you should specify the USRP option. You will be prompted to enter a list of event identifiers (EIDs). The EIDs used by WebSphere MQ are:

- ▶ *5E9* To collect information about control blocks on entry to WebSphere MQ
- ▶ *5EA* To collect information about control blocks on exit from WebSphere MQ
- ▶ *5EB* To collect information on entry to internal WebSphere MQ functions
- ▶ *5EE* To collect information on exit from internal WebSphere MQ functions
- ▶ *FC6* To collect information about CICS/MQ functions

You can also use the JOBNAMEP option, specifying the batch, CICS, IMS, or TSO job name, to limit the trace output to certain jobs.

Once started, you can display information about, alter the properties of, and stop the trace with the **DISPLAY TRACE**, **ALTER TRACE**, and **STOP TRACE** commands.

To use any of the trace commands, you must have one of the following:

- ▶ Authority to issue **START/STOP TRACE** commands (trace authority)
- ▶ Authority to issue the **DISPLAY TRACE** command (display authority)

To format the user parameter data collected by the global trace you can use the **IPCS GTFTRACE USR(yyy)** command, where **yyy** is:

5E9	To format information about control blocks on entry to WebSphere MQ MQI calls
5EA	To format information about control blocks on exit from WebSphere MQ MQI calls
5E9,5EA	To format information about control blocks on entry to and exit from WebSphere MQ MQI calls
5EB,5EE	To format information about control blocks on entry to and exit from WebSphere MQ internal functions

The key pieces of diagnostic data required for diagnosing WebSphere MQ on z/OS problems are the MQ MSTR and CHIN (Channel Initiator) job logs, the

CHIN trace for remote messaging activity and the MSTR trace which is processed via GTF (Generalised Trace Facility). The CHIN trace data is always written to the CHIN dataspace and must be captured during the DUMP process.

Some key parameters must be set up to ensure sufficient space is allocated within the WebSphere MQ address spaces to hold the required data. These can be set up as follows:

- ▶ Set the CHIN trace table size via **CSQ6CHIP** to 5 MB.
- ▶ Set the trace table size via **CSQ6SYSP** to 250.
- ▶ MQ needs to be recycled to pick up these changes.
- ▶ Start GTF Trace using **PARM=MODE (INT)** with REGION=1000K

The GTF parameters SADMP, SDUMP and UBDUMP should all be set to at least 10M.

- ▶ **TRACE=JOBNAMEP,USRP**
- ▶ **JOBNAME=(ssidMSTR,jobname),USR=(5E9,5EA,5EB,5EE,FC6)**

Now that the GTF trace is running, the internal MQ trace is started to output the MQ trace data to the GTF.

- ▶ Start MQ tracing as follows:

```
+cpfSTART TRACE(GLOBAL) DEST(GTF) CLASS(*) RMID(*)
```

Although you have specified DEST(GTF), CHIN trace data will still be routed to DEST(RES). Everything else will go to GTF, which will be dumped when the MSTR, CHIN and CHIN dataspace are dumped, due to MODE (INT) GTF.

It might be a good idea to **SLIP** on the CSQXxxxE message if this is the common point of failure.

```
SLIP SET,IF,LPAMOD=(IGC0003E,0),  
DATA=(1R?+4,EQ,C3E2D8E7,1R?+8,EQ,FxFxFxC5),  
JOBNAME=ssidCHIN,  
JOBLIST=(ssidMSTR,ssidCHIN),  
DSPNAME=('ssidCHIN'.CSQXTRDS),  
SDATA=(CSA,RGN,PSA,SQA,LSQA,TRT,SUM),  
MATCHLIM=1,END
```

or you can use the more streamlined **MSG SLIP** as follows:

```
SLIP SET,MSGID=CSQXxxxE,  
JOBNAME=ssidCHIN,  
JOBLIST=(ssidMSTR,ssidCHIN),  
DSPNAME=('ssidCHIN'.CSQXTRDS),  
SDATA=(CSA,RGN,PSA,SQA,LSQA,TRT,SUM),  
MATCHLIM=1,END
```

This will dump QMGR and CHIN and Dataspace when msgid CSQXxxxE is generated by the MQ CHIN address space. Naturally, you have to substitute a valid error message.

- ▶ To stop MQ Internal trace:

```
+cpfSTOP TRACE(*) **** +cpf = ssid prefix ****
```

```
STOP GTF
```

The following shows a sample MQ 5EA trace entry.

```
USRD9 5EA ASCB 00F31200          JOBN MQ69CHIN
CSQW073I EXIT: MQSeries user parameter trace
OPEN
Thread... 008D7838  Userid... COAKLEY  pObjDesc. 0EBB2F38
RSV1..... 00000000  RSV2..... 00000000  RSV3..... 00000000
CompCode. 00000000  Reason... 00000000
D4D8F6F9  C3C8C9D5  00000000  00000000  | MQ69CHIN..... |
E3F0F0F8  C4F7F8F3  F8404040  40404040  | T008D7838      |
40404040  40404040  40404040  40404040  |                  |
40404040  40404040  40404040  40404040  |                  |
USRD9 5EA ASCB 00F31200          JOBN MQ69CHIN
CSQW073I EXIT: MQSeries user parameter trace
+0000 D6C44040  00000001  00000005  00000000  | OD ..... |
+0010 00000000  00000000  00000000  00000000  | ..... |
+0020 00000000  00000000  00000000  00000000  | ..... |
+0030 00000000  00000000  00000000  D4D8F6F9  | .....MQ69 |
+0040 40404040  40404040  40404040  40404040  | ..... |
+0050 40404040  40404040  40404040  40404040  | ..... |
+0060 40404040  40404040  40404040  40404040  | .....CSQ. |
+0070 5C000000  00000000  00000000  00000000  | *..... |
+0080 00000000  00000000  00000000  00000000  | ..... |
```

WebSphere MQ channel trace

It is worth noting that a wrap-around line trace is always active for each channel and the trace data is stored in a 4K buffer (one for each channel) in the CHIN address space.

This is a good source of data for reviewing problems with LU6.2 and TCP/IP channels.

To allow the trace data to be reviewed, you must have a CSQSNAP DD statement in your CHIN JCL. To display (write it to the dataset specified in the CSQSNAP DD, for example SYSOUT=*) the trace data, you must start a CHIN trace using **START TRACE(GLOBAL) RMID(231) CLASS(4) IFCID(202)** and then

DISPLAY CHSTATUS(channel) CURRENT for the relevant channel you need to trace.

IPCS and WebSphere MQ

WebSphere MQ for z/OS provides a set of panels to help you process dumps using IPCS.

1. From the IPCS Primary Option menu, select **ANALYSIS (Analyze dump contents)**. The IPCS MVS Analysis Of Dump Contents panel appears.
2. Select **COMPONENT - MVS component data** (Option 6). The IPCS MVS Dump Component Data Analysis panel appears. Its appearance depends on the products installed, but will be similar to that shown below.

```
----- IPCS MVS DUMP COMPONENT DATA ANALYSIS -----
OPTION ==> SCROLL ==> CSR
To display information, specify S option name or enter S to the
left of the Option desired. Enter ? to the left of an Option to
display help regarding the component support.
Name Abstract
ALCWAIT Allocation wait summary
AOMDATA AOM analysis
ASMCHECK Auxiliary storage paging activity
ASMDATA ASM control block analysis
AVMDATA AVM control block analysis
COMCHECK Operator communications data
CSQMAIN MQSeries dump formatter panel interface
CSQWDMQ MQSeries dump formatter
CTRACE Component trace summary
DAEDATA DAE header data
DIVDATA Data-in-virtual storage
```

3. Select the **CSQMAIN** WebSphere MQ dump formatter panel interface.
4. The WebSphere MQ - Dump Analysis menu appears as shown next.

```
-----IBM MQSeries for MVS/ESA - DUMP ANALYSIS-----
COMMAND ==>
1 Display all dump titles 00 through 99
2 Manage the dump inventory
3 Select a dump
4 Display address spaces active at time of dump
5 Display the symptom string
6 Display the symptom string and other related data
7 Display LOGREC data from the buffer in the dump
8 Format and display the dump
9 Issue IPCS command or CLIST
```

You can also use IPCS to interrogate the dump using the WebSphere MQ **VERBX CSQWDMP** command or the Channel Initiator **VERBX CSQXDPRD**. For example:

- ▶ For default formatting of all address spaces, using information from the summary portion of the dump use:

```
VERBX CSQWDMP
```

- ▶ To display the trace table from a dump of subsystem named MQMT, which was initiated by an operator (and so does not have a summary portion) use:

```
VERBX CSQWDMP 'TT,SUBSYS=MQMT'
```

- ▶ To display all the control blocks and the trace table from a dump produced by a subsystem abend, for an address space with ASID (address space identifier) 1F, use:

```
VERBX CSQWDMP 'TT,LG,SA=1F'
```

- ▶ To display the portion of the trace table from a dump associated with a particular EB thread, use:

```
VERBX CSQWDMP 'TT,EB=nnnnnnnn'
```

- ▶ To display message manager 1 report for local non-shared queue objects whose name begins with ABC, use:

```
VERBX CSQWDMP 'MMC=1,ONAM=ABC,Obj=MQLO'
```

IPCS VERBEXIT CSQXDPRD enables you to format a channel initiator dump. You can select the data that is formatted by specifying keywords. For example:

```
VERBX CSQXDPRD 'SUBSYS=MQMT,CHST=3'
```

This will display all channel information, a program trace, line trace and formatted semaphore table print of all channels in the dump.

WebSphere Application Server: z/OS

This section describes some of the key areas of diagnostic data that can be used for Problem Determination (PD) in WebSphere Application Server on z/OS. As with many of the products that are built using numerous components and utilize a broad range of system functions, collecting diagnostic data from all the related components in many problem scenarios is not mandatory. Many problems will require only specific diagnostic data to be collected for the individual component that is experiencing the problem, whereas the more complex problems will require a larger subset of data to be collected, or even diagnostic data collection for all related components.

We will discuss the diagnostic processes that will assist the system programmer with collecting data to assist with WebSphere problem diagnosis.

Diagnostic data

The WebSphere for z/OS error log is a logstream data set managed by the z/OS System Logger. This log stream resides on either a staging data set on DASD or in the Coupling Facility.

The data stored in the WebSphere Application Server log stream must be formatted using the BBORBLOG Rexx Exec routine. By default, BBORBLOG formats the error records to fit a 3270 display.

```
LOGSTREAMNAME=<LOG_STREAM_NAME>
```

where the LOG_STREAM_NAME is a valid MVS data set name with no more than 26 characters. An example is WAS.ERROR.LOG.

During WebSphere Application Server initialization, an attempt is made to connect to the appropriate logstream data set. If this connection is successful, you will see the following message, which indicates the name of the data set being used:

```
BOU0025I ERRORS WILL BE WRITTEN TO <logstream name> LOG STREAM FOR JOB  
<server name>
```

If, however, the server cannot connect to the logstream, the message is instead written to CERR, which puts it in the SYSOUT of the job output. This would be indicated by the message:

```
BBU0025I ERRORS WILL BE WRITTEN TO CERR FOR JOB <server name>
```

The log stream records error information when WebSphere for z/OS detects an unexpected condition or failure within its own code. Use the error log stream in conjunction with other facilities available to capture error or status information, such as an activity log, trace data, system logrec, and job log.

A sample from the WebSphere for z/OS log might include:

```
2002/09/05 20:26:10.233 01 SYSTEM=SC49 SERVER=NAMING01 JobName=BBONMS  
ASID=0X0060 PID=0X01060042 TID=0X23AE32E0 0X000008 c=1.19  
./bboi3pli.cpp+3712 ... BBOU0011W The function  
IB0IM390PrivateLocalToServer_IMContainer_Impl::beforeMethodDispatch(::  
ByteString*,CORBA::Object_LocalProxy_ptr,const  
char*,CORBA::Long)+3712 raised CORBA system exception  
CORBA::INV_OBJREF. Error code is C9C21444
```

Joblog and SYSLOG

Each MVS address space maintains a number of JES (Job Entry Subsystem) output files that contain information related to the state of the control and server regions.

The SYSLOG (System Log) stores messages that are written to the master console. Most serious messages related to WebSphere Application Server on z/OS will be written to both the syslog and also to the WebSphere joblogs.

The output from both the JOBLLOG and SYSLOG can be reviewed using **SP00L DISPLAY** and **SEARCH FACILITY** (SDSF). The key JES Spool DD names associated with WebSphere address space problem diagnosis are as follows.

JESMSG LG - This section contains start-up messages, including a list of environment variable values and server settings, and the service level of WebSphere:

```
BB0U0245I CURRENT CB SERVICE LEVEL IS build level W401078 release.
```

It also lists the Java service level when in a J2EE server region:

```
BB0J0011I JVM Build is "J2RE 1.3.1 IBM OS/390 Persistent Reusable VM.  
build hm131s-20020723 (JIT enabled: jitc)".
```

JESYSMSG - This section may list more messages, dump information and, again, a list of environment variables and server settings.

CEEDUMP - An exception in the address space may cause this section to be generated. It lists failure information including trace backs (a *trace back* shows which functions were last called prior to the program failure).

SYSOUT - During normal processing, the SYSOUT should be empty, but there are situations that cause output to be written to this section. If the error log logstream cannot connect, then the messages set to be written to the error log will be written to CERR, which goes to SYSOUT. Trace from the JVM when you set the environment variable JVM_DEBUG=1 and jvm_logfile is not set.

SYSPRINT - The WebSphere for z/OS trace output can be written to SYSPRINT if the environment variable TRACEBUFFLOC=SYSPRINT. If LDAP_DEBUG=65535 is set, then LDAP trace is written here for the naming server region.

The JES spool writes the information to the job logs of the address spaces for the different regions. There are several types of regions that are interesting:

- ▶ For each server instance, there is one control region and zero or more server regions (except for the DAEMON, for which there is only a control region).
- ▶ Control regions, to put it simply, handle communication, receiving requests from clients and sending back responses.
- ▶ Server regions are given the requests to process, so they do the actual work.
- ▶ There is also the base set of server instances, as well as the J2EE server instances.

- ▶ You may also have address spaces from local clients, the LDAP server, and your HTTP Server.

CTRACE (SYSBBOSS)

WebSphere for z/OS uses the CTRACE component SYSBBOSS to capture and display trace data in trace data sets. You can send the CTRACE output to the spool (SYSPRINT) or to a dataset. It is configured by the parameter TRACEBUFFLOC in the current.env file. We suggest that while you are running the bootstrap process, you set TRACEBUFFLOC=SYSPRINT (if you set it in /WebSphere390/CB390/<plexname>/initial/configuration.env, it will get propagated to all the servers).

Once you move beyond the IVP, change it back to BUFFER. Make sure you have the CTRACE writer configured before switching.

Executing the CTRACE for WebSphere

Follow these steps to obtain a formatted CTRACE that is prepared for debugging.

1. Planning for component tracing

- Create CTIBBOxx PARMLIB members for WebSphere for z/OS tracing. This is used only when TRACEBUFFLOC=BUFFER was specified:

WebSphere for z/OS provides a default CTRACE PARMLIB member, in CTIBBO00. Following is the format of this member:

```
TRACEOPTS
/* Start a ctrace writer. Remove comments to start the PROC */
/* during CB Series address space initialization. */
WTRSTART(BBOWTR)
/* Indicate that tracing is active for CB Series: */
ON
/* Connect to ctrace external writer (BBOWTR): */
WTR(BBOWTR)
```

Note: You still have to start the BBOWTR started task yourself; WebSphere will not start it.

- Specify buffers for WebSphere for z/OS tracing

WebSphere for z/OS enables you to specify two buffer options at process (address space) initialization. These options control the number of buffers and the size of each buffer. They are environment variables, and must be specified in the current.env file.

TRACEBUFFCOUNT=nnnn

This specifies the number of trace buffers to allocate. The default is 4.

TRACEBUFFSIZE=nnnn

This specifies the size of a single trace buffer in bytes. The default of 1MB.

TRACEBUFFLOC=SYSPRINT | BUFFER

This specifies where you want trace records to go: either to SYSPRINT or to a memory buffer (BUFFER), then to a CTRACE data set. The default is to direct trace records to SYSPRINT for the client, and to a buffer for all other WebSphere for z/OS processes.

- Specify WebSphere for z/OS trace options through environment variables (PARMLIB override)

In addition to the options that you can specify in the CTIBBOxx PARMLIB member at process (address space) initialization, you can use environmental variables with additional options that enable you to establish levels of tracing and PARMLIB override. These parameters must be specified in the current.env file:

- TRACEALL=x

This establishes the general trace level for all WebSphere for z/OS components. Valid trace levels are 0 (none), 1 (exception), 2 (basic), and 3 (detailed tracing). Use 1 unless you have a problem; if there is a problem, use 3.

- TRACEPARAM=pp

This specifies the PARMLIB member that contains the WebSphere for z/OS CTRACE connection and startup information. pp is either a two-character suffix to be added to CTIBBO to form CTIBBOpp, or it is a fully specified name of a member of PARMLIB. The default is 00. This parameter is valid only for the daemon address space, and must be specified as a program environment variable.

2. Obtaining the trace data

- a. You first need to allocate the trace data set (DCB: PS, VB, 32756, 32760). Start the CTRACE writer address space. This is automatically done in the default WebSphere for z/OS PARMLIB member CTIBBO00. If the CTRACE writer was not started, you can do it now with the command:

```
TRACE CT,WTRSTART=BBOWTR
```

You will see this message:

```
ITT110I INITIALIZATION OF TRACE WRITER BBOWTR COMPLETE.
```

- b. Start the daemon address space with the desired trace specifications.
- c. Start the WebSphere for z/OS servers.

- d. If you need to collect trace data for problem analysis, do the following:
- i. Disconnect WebSphere for z/OS from CTRACE by using the operator command:


```
TRACE CT,ON,COMP=SYSBBOSS
REPLY x,WTR=DISCONNECT,END
```
 - ii. Stop the CTRACE address space by using the operator command:


```
TRACE CT,WTRSTOP=BBOWTR
```

Example A-1 shows the syslog with the commands to disconnect WebSphere for z/OS from CTRACE, and to stop the CTRACE address space.

Example: A-1 Disconnecting from CTRACE and stopping CTRACE address space

```
SY1      TRACE CT,ON,COMP=SYSBBOSS
SY1      *04 ITT006A SPECIFY OPERAND(S) FOR TRACE CT COMMAND.
SY1      REPLY 4,WTR=DISCONNECT,END
SY1      IEE600I REPLY TO 04 IS;WTR=DISCONNECT,END
SY1      ITT038I ALL OF THE TRANSACTIONS REQUESTED VIA THE TRACE CT
          COMMAND WERE SUCCESSFULLY EXECUTED.
SY1      IEE839I ST=(ON,0064K,00128K) AS=ON BR=OFF EX=ON MT=(ON,016K)
          ISSUE DISPLAY TRACE CMD FOR SYSTEM AND COMPONENT TRACE STATUS
          ISSUE DISPLAY TRACE,TT CMD FOR TRANSACTION TRACE STATUS
SY1      TRACE CT,WTRSTOP=BBOWTR
SY1      ITT038I ALL OF THE TRANSACTIONS REQUESTED VIA THE TRACE CT
          COMMAND WERE SUCCESSFULLY EXECUTED.
SY1      IEE839I ST=(ON,0064K,00128K) AS=ON BR=OFF EX=ON MT=(ON,016K)
          ISSUE DISPLAY TRACE CMD FOR SYSTEM AND COMPONENT TRACE STATUS
          ISSUE DISPLAY TRACE,TT CMD FOR TRANSACTION TRACE STATUS
SY1      AHL904I THE FOLLOWING TRACE DATASETS CONTAIN TRACE DATA :
          KEITHK.BOSS.CTRACE
SY1      ITT111I CTRACE WRITER BBOWTR TERMINATED BECAUSE OF A WTRSTOP
          REQUEST.
```

Note: We recommend that you disable CTRACE for better performance. If IBM Support asks for a CTRACE, you can enable it at that time.

See *WebSphere Application Server for z/OS V4.0.1: Messages and Diagnosis*, GA22-7837 for more information.

3. Viewing the CTRACE with the IPCS

From the Interactive Problem Control Facility (IPCS) Primary Option Menu panel, select option **0 (DEFAULTS)** and enter the name of the trace data set,

then click **Intro**. Now, you have to issue the following IPCS command from the option 6 (COMMAND):

```
CTRACE COMP(SYSBBOSS)FULL
```

If you are interested in only JRas data, enter the following IPCS command:

```
CTRACE COMP(SYSBBOSS)USEREXIT(JRAS)
```

MVS Interactive Problem Control System (IPCS) Commands, SA22-7594, describes how to use IPCS CTRACE, and contains a complete list of **CTRACE** command parameters.

To send this output to a dataset, you can use IPCS commands. The following steps describe the process:

- From any IPCS command line, execute these commands:

```
TSO ALLOC FI(IPCSPRNT) SYSOUT(H)
TSO ALLOC FI(IPCSTOC) SYSOUT(H)
```

- You need to have PRINT specified as one of the defaults:

```
IP SETEDF PRINT NOTERM
```

This causes output to only go to PRINT, and not to the screen (you can use TERM instead, but then you have scroll to the bottom of the output display to get all the output in the PRINT file).

- When IPCSPRNT is used as the DDNAME (as in this case), then the first command issued that causes output will open the print file:

```
IP CBF CVT
```

This IPCS command is to format the CVT.

- Now you can format things and run execs, then close the print file:

```
IP CLOSE PRINT
```

- You can now go to another ISPF window, get into SDSF, and put a question mark (?) next to your active user ID. IPCSPRNT should be one of the DDNAMES present. Your output (for example, the preceding formatted CVT) should be there.

You can now use the **XDC** command (put xdc next to your active user ID) to copy this output to a dataset.

- You need to free the files you allocated with the following IPCS commands:

```
TSO FREE FI(IPCSPRNT)
TSO FREE FI(IPCSTOC)
```

Note: If, after these steps, you leave IPCS, then come back and try to issue an IPCS command, you will receive the message: Unable to open PRINT FILE(IPCSPRNT). If this occurs, go into IPCS option 0 for DEFAULTS and change PRINT to NOPRINT.

WebSphere Business Integrator: z/OS

In this section, we will discuss the components that make up the WebSphere Business Integrator environment, then we will look at the sources of diagnostic data.

Components of WebSphere Business Integrator on z/OS

The WebSphere Business Integrator architecture manages the modeling of the application connectivities and integration operations using message flows and nodes, and the modeling of the message structure to enable processing within these nodes.

The *Broker* is the runtime environment that assigns units of execution (threads) to message flows to process messages. It takes the models and realizes them as concrete implementations. The primary objective of the broker is to achieve consistency at the application layer. This means that a message flow or set modeled in the Control Center and deployed via the Configuration Manager must operate without reference to the platform of execution. So, for example, we do not want to have to have ESQL that has distinct processing for z/OS.

The modeling of the message flows and messages is performed using a graphical user interface called the *Control Center*. It also allows modeling of broker topologies and assignment of resources (message flows and sets) to these topologies. There is support for management of publish/subscribe topics and the Access Control Lists (ACLs) governing their security.

Resources created and manipulated by the Control Center are stored in databases controlled by the Configuration Manager (DB2 on the z/OS platform). This is the master repository for WebSphere Business Integrator resources. An authorized Control Center operator can deploy resources to the broker for runtime use.

Finally, the User Name Server is an optional component used to support publish/subscribe ACL processing. It needs to operate on any platform where it can gather relevant principals (users and groups) and report them to the Broker and Configuration Manager.

The broker runtime environment is a “collection” of address spaces (ASIDs), which allows natural isolation, recovery and scalability. Each address space contains at least two Language Environment (LE) processes. The first, or “infrastructure” process, is started as an authorized process so that it can create z/OS components, for example, Program Calls (PCs) for SVC dumps, etc., and then return to problem state. This process only exists on z/OS.

After initialization, it creates and monitors a second process, which performs the main brokering function. The main process in each ASID runs platform-independent code using C++ and Java (publish/subscribe) to implement the brokering function.

The brokering function address spaces are as follows.

Control Process

This is the broker started task address space. It is small, being a monitor for failures of the Administration Agent (AA) address spaces. On z/OS, a console listener thread enables z/OS console interactions with users through the MODIFY interface.

Administration Agent

This serves as the administration to the Configuration Manager and, by extension, the command center. It manages the deployment of message flows and message sets, and manages the lifecycle and command reporting of execution groups (EG).

Execution Group

This is where the message flows deployed from the Configuration Manager execute. The DataFlowEngine process itself contains a number of threads and predefined flows (Configuration, PubSub Control) to support the various brokering functions. Multiple EG address spaces remove any concern about (Virtual Storage Constraint Relief) VSCR.

User Name Server

This address space retrieves all valid principals (users, groups) from z/OS and reports them to the Configuration Manager (CM) and broker to support publish/subscribe Topic Access Control List (ACL) processing. It has its own Control Process.

User Process

User commands can be issued not only from the console but from JCL and directly through a UNIX Shell.

There are a large number of associated address spaces with which the broker interacts.

OMVS

This address space provides several industry standard interfaces (XPG4) that allow the WebSphere Business Integration processing model and code to be largely platform-independent.

WebSphere MQ

This is one of the primary transports for dataflows, and WebSphere Business Integration uses it for inter-process and inter-platform communication. For example, the AA communicates with the EGs and CM using XML messages flowed over WebSphere MQ.

DB2

Again, this is heavily used by dataflows for data warehousing and augmentation, but it is also used to store the deployed dataflows and message dictionaries. The broker also keeps some of its internal state in DB2.

RRS

Since the broker runs within regular z/OS address spaces, Resource Recovery Services (RRS) are the transaction manager that enables the coordination of resources (message queues, database tables) accessed by a dataflow.

Capturing a dump of the relevant address spaces

As you can see from the previous WebSphere Business Integrator components and related products, collecting diagnostic data can be a complex process. In most cases, we will be dumping multiple address spaces which will probably include the Control Process address space, the Administration Agent, the Execution Group(s), and possibly the Unix Systems Services address space, and even DB2 or RRS.

The key requirements are to dump the Control Process (usually with a name that includes the BRK suffix, for example, MQ01BRK), the Administration Agent (with a suffix of BRK1, for example, MQ01BRK1) and the failing execution group that will have a suffix that starts with BRK2 for the first execution group address space and increments for each additional execution group, for example, MQ01BRK2, MQ01BRK3, MQ01BRK4, etc.

It may also be necessary to include the WebSphere MQ Queue Manager address space in the dump.

For example:

```
DUMP COMM=(WBI CP/Broker/Execution Dump)
*01 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
R 01,JOBNAME=(MQ01BRK,MQ01BRK1,MQ01BRK2,USSstc, MQ01MSTR),CONT
*02 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
IEE600I REPLY TO 01
IS;JOBNAME=(MQ01BRK,MQ01BRK1,MQ01BRK2,USSstc,MQ01MSTR),CONT
R 02,SDATA=(CSA,RGN,PSA,SQA,LSQA,TRT,SUM),END
IEE600I REPLY TO 02 IS;SDATA=(CSA,RGN,PSA,SQA,LSQA,TRT,SUM),END
IEA794I SVC DUMP HAS CAPTURED: 869
DUMPID=001 REQUESTED BY JOB (*MASTER*)
```

Displaying the status of a trace

To display the status of a trace, use the report trace command, **reporttrace (RT)**.

Note that names are case sensitive and that you should include the names in single quotes if they contain mixed case characters.

For example, this command:

```
F MQP1BRK,RT T=YES,E='FONEZONE ',F='MFFONEZONE '
```

produces the following output:

```
+BIP8098I MQP1BRK Trace level:none,mode:safe,size:4096 KB.
+BIP8071I MQP1BRK Successful command completion.
```

Collecting a user execution group trace

To collect a user execution group trace, use the collect trace command, **changetrace (CT - Change trace settings)**. If your flow name is in mixed case, you need to enclose it in single quotes.

For example, to collect a trace, turn trace on using the command:

```
F broker,CT,u=YES,L=debug,E='default '
```

When you have captured your trace, turn trace off with:

```
F broker,CT,u=YES,L=none,E='default '
```

To format your trace, do the following:

1. Edit your component PDSE, and create a new member, for example EGTRACE .

Copy in BIPJLOG.

2. Search for `-t -b agent` and change it to `-u -e default`, where `default` is the name of your execution group.
3. Search for `format.log`. This is where the output file is stored. Change the name if you want to have trace-specific names.

Collecting a service execution group trace

IBM might ask you to collect a trace of the internal processing of an execution group.

To turn tracing on, use the collect trace command, **changetrace** (CT - Change trace settings). If your flow name is in mixed case, you need to enclose it in single quotes.

For example to collect a trace, turn trace on using the command:

```
F broker,ct t=YES,L=debug,E='default '
```

When you have captured your trace, turn trace off with:

```
F broker,ct t=YES,L=none,E='default '
```

To format your trace:

1. Edit your component PDSE, and create a new member, for example `SEGTRACE`.
Copy in `BIPJLOG`.
2. Search for `-t -b agent` and change it to `-t -e default`, where `default` is the name of your execution group.
3. Search for `format.log`. This is where the output file is stored. Change the name if you want to have trace-specific names.

Collecting a WebSphere Business Integrator service trace

IBM might ask you to collect a trace of the internal processing, not related to an execution group. To turn trace on, use the **changetrace** command. If your flow name is in mixed case, you need to enclose it in single quotes. For example, to collect a trace, turn trace on using the command:

```
F broker,ct t=YES,L=debug,b=yes
```

When you have captured your trace, use the command:

```
F broker,ct t=YES,L=none,b=yes
```

To format your trace:

1. Edit your component PDSE, and create a new member, for example STRACE.
Copy in BIPJLOG.
2. Search for format.log. This is where the output file is stored. Change the name if you want to have trace-specific names.

Checking for dumps and other diagnostic information

You should check for dumps and other diagnostic information. Information is stored in the following places:

- ▶ In the home directory of the started task user ID. An authorized user can issue the TSO command `LU id OMVS` to display this information.
- ▶ In the component directory in the /output subdirectory.

For example, this could be found for a broker called MQP1BRK in the file /var/wmqi/MQP1BRK/output.

Logs

The following section contains information about Syslog and job logs.

Job log and Syslog

SYSLOG (SDSF) contains all BIP messages. In the event of a problem, it is also worth looking for messages from other subsystems that may be a factor in the WebSphere Business Integration problem, for example, DB2, WebSphere MQ or RRS.

Because of the way the WebSphere Business Integration Administrative Agent (AA) and Execution Group (EG) address spaces are started, their messages are not collected under their job logs. You can also review the SYSLOGD HFS file (if you are using this).

To overcome this difficulty, the SYSLOGD allows you to route joblog messages to a joblog file defined by rules in /etc/syslog.conf.

Useful HFS files

- ▶ *.../output/stderr, .../output/stdout* contain minimal output from WQMI, including JVM.
- ▶ *.../output/cvpplog, .../output/cvpserr, .../output/cvpslog* contain the output from the customization verification job.

- ▶ *.../output/joberr, .../output/jobout* contain output from customization jobs BIP\$DB01-05, BIP\$MQ01 and BIP\$UT01.
- ▶ *.../output/traceodbc* contains ODBC tracing information. Enable this through DSNAOINI.
- ▶ *.../ENVFILE* contains USS settings for the broker or user name server.
- ▶ *.../dsnaoini* contains ODBC configuration information.
- ▶ *.../mqsicompCIF* contains the master source for the WebSphere Business Integration configuration.

Trace files

Examples of the file name format for various WebSphere Business Integration traces are as follows.

- ▶ **Administration Agent service trace** *.../log/<broker name>.agent.trace.bin.0*
- ▶ **Administration Agent user trace** *.../log/<broker name>.agent.userTrace.bin.0*
- ▶ **WBI command service trace** *.../log/<broker name>.mqsistop.trace.bin.0*
- ▶ **WBI command user trace** *.../log/<broker name>.mqsistop.userTrace.bin.0*
- ▶ **Control Process service trace** *.../log/<broker name>.service.trace.bin.0*
- ▶ **Control Process user trace** *.../log/<broker name>.service.userTrace.bin.0*
- ▶ **Execution Group/DataFlow service trace** *.../log/<broker name>.02345678-1234-1234-1234-123456789003.trace.bin.0*
- ▶ **Execution Group/DataFlow user trace** *.../log/<broker name>.02345678-1234-1234-1234-123456789003.userTrace.bin.0*

Trace files should be formatted at the customer site using BIPJLOG JCL in component PDS.

LDAP: z/OS

For WebSphere for z/OS, the Lightweight Directory Access Protocol (LDAP) component of the z/OS Secureway Security Server provides the directory services for the Java Naming and Directory Interface (JNDI), CORBA (MOFW) naming and interface repository services. The contents of the directory are stored in DB2 tables.

Like other z/OS components, the LDAP has a trace that is very useful when you encounter problems related to security and authorization issues.

The LDAP trace is started and stopped dynamically. You can also activate it when the LDAP starts. The output goes directly to the joblog of the LDAP started task.

Starting an LDAP trace

When the LDAP server is running as a started task or from the OS/390® shell, it is possible to dynamically turn the debugging facility on and off. The following command can be sent to the LDAP server from SDSF or from the operator console. Note that if the command is entered from SDSF, it must be preceded by a slashmark (/). The command is:

```
f ldapsrv,appl=debug=nnnn
```

where nnnn is the decimal value of the desired debug level.

Note: To send the same command to the LDAP server in the OS/390 shell, you need to know the job name assigned to the process. To find out this job name, enter the following command from SDAF or from the operator's console:

```
d a,1
```

Once you find out the job name (which includes the user ID under which the LDAP server is running, and a suffix), use it to replace **ldapsrv** in the preceding command.

The most used debug level values are 1114111 (which traces everything for the TDBM back end), and 65535 (for the RDBM back end), but a list of all debug levels is provided in *SecureWay Security Server LDAP Server Administration and Usage Guide*, SC24-5923.

Debug information will be added to the job log output associated with the LDAP server.

When you turn on the trace, the following message should appear in the console:

```
GLD0091I Successfully set debug level to 65535 from console command.
```

To turn debug tracing off, enter the same command, but providing the value zero (0) for nnnn; this message will appear:

```
GLD0091I Successfully set debug level to 0 from console command.
```

For problems starting the LDAP server, you can set up the trace at initial start by using the following command:

```
s ldapsrv -d nnnn
```

If you have problems in the bootstrap CosNaming, you can specify the following parameters in the current.env file:

```
LDAP_DEBUG=65535
TRACEBUFFLOC=BUFFER SYSPRINT
```

For more information about the LDAP trace, see *SecureWay Security Server LDAP Server Administration and Usage Guide*, SC24-5923.

IBM HTTP Server

This section provides detailed information for IBM HTTP Server problem determination on different platforms.

General configuration

The following list of files are needed for debugging:

- ▶ IBM HTTP Server version: use the following command to display the full IHS version.

Windows (1.3.12x, 1.3.19x, 1.3.26x, 2.0.42x): <IHS_root>/apache -v

Unix 1.3.12x, 1.3.19x, 1.3.26x): <IHS_root>/bin/httpd -ver

Unix (2.0.42x): <IHS_root>/bin/apachectl -V

- ▶ Configuration file: <IHS_root>/conf/httpd.conf

- ▶ Error log:

Windows: <IHS_root>/logs/error.log

Unix: <IHS_root>/logs/error_log

- ▶ Access log:

Windows: <IHS_root>/logs/access.log

Unix: <IHS_root>/logs/access_log

Global Security Kit (GSKit)

If you have to deal with security issues, you will also need the following information for debugging purposes.

- ▶ Version 1.3.12x
 - Windows: /program files/ibm/gsk4/bin/gsk4ver.exe
 - AIX - /usr/opt/ibm/gskit/bin/gsk4ver
 - SUN - /opt/ibm/gsk4/bin/gsk4ver
 - HP - /opt/ibm/gsk4/bin/gsk4ver

- LINUX - /usr/local/ibm/gsk4/bin/gsk4ver
- ▶ Version 1.3.19x, 1.3.26x, 2.0.42x
 - Windows: /program files/ibm/gsk5/bin/gsk5ver.exe
 - AIX - /usr/opt/ibm/gskkm/bin/gsk5ver
 - SUN - /opt/ibm/gsk5/bin/gsk5ver
 - HP - /opt/ibm/gsk5/bin/gsk5ver
 - LINUX - /usr/local/ibm/gsk5/bin/gsk5ver

SSL handshake and configuration issues

For debugging, you will need all the files from “General configuration” on page 355 in addition to the following information.

GSKit and SSL (IHS stand-alone)

1. Stop IHS.
2. Clear all logs in the <IHS_root>/logs directory.
3. Edit the httpd.conf:
 - a. Change LogLevel to debug.
 - b. Add the SSLTrace directive to the bottom of the httpd.conf file.
4. Enable GSKIT trace:
 - Windows: Create a system variable called: GSK_TRACE_FILE Set the value with the name for the log file. For example: c:\gsktrace.log.
 - Unix: As the user ID that starts the IBM HTTP Server, create an environment variable called: GSK_TRACE_FILE. The environment variable can be created in either of two ways:
 - Using the **setenv GSK_TRACE_FILE=value** (full path and filename), for example in csh: **setenv GSK_TRACE_FILE /usr/HTTPServer/logs/gsktrace_log**
 - Exporting **GSK_TRACE_FILE=value** (full path and filename), for example in ksh: **export GSK_TRACE_FILE=/usr/HTTPServer/logs/gsktrace_log**
5. Start IHS.
6. Recreate the problem.
7. Capture activity with the command: **netstat -na > netstat.out**.
8. Provide the following datafiles for review:
 - a. httpd.conf, error_log, access_log
 - b. netstat.out
 - c. gsktrace_log

Also include the date/time of failure along with the browser version and full URL (for example: <https://www.mycompany.com/mystuff/goodies/index.html>) that resulted in the SSL failure.

GSKit and SSL (IHS with WebSphere)

1. Stop IHS and WebSphere.
2. Clear all logs in the <IHS_root>/logs directory.
3. Clear all logs in the <WebSphere_root>/logs directory.
4. Edit the plugin-cfg.xml and change Loglevel to Trace (Plugin Trace), for example: `<Log LogLevel="Trace" Name="/path/to/logs/native.log"/>`
5. Edit httpd.conf.
 - Change Loglevel to debug.
 - Add the SSLTrace directive to the bottom of the httpd.conf file.
6. Enable GSKIT trace:
 - Windows: Create a system variable called: GSK_TRACE_FILE, set the value with the name for the log file.
 - UNIX: As the user ID that starts the IBM HTTP Server, create an environment variable called: GSK_TRACE_FILE.
7. Restart IHS and WebSphere.
8. Recreate the problem.
9. Capture the activity with `netstat -na > netstat.out`.
10. Provide the following datafiles for review.
 - a. httpd.conf, error_log, access_log
 - b. plugin-cfg.xml
 - c. native.log (4.0x) http_plugin.log (5.0x)
 - d. stderr and stdout
 - e. netstat.out
 - f. gsktrace_log

Also include the date/time of failure along with the browser version and full URL (for example: <https://www.mycompany.com/mystuff/goodies/index.jsp>) that resulted in the SSL failure.

CMS key database (.kdb) and certificate issues

For debugging, you will need all the files from “General configuration” on page 355 in addition to the following information.

1. CMS key database file (.kdb): the CMS Key database file is defined by the KeyFile directive in the httpd.conf: KeyFile /<key_file_path>/key.kdb.
Please provide the password also if possible.
2. Password stashfile (.sth), for example: /usr/HTTPServer/ssl/key.sth
3. request database (.rdb), for example: /usr/HTTPServer/ssl/key.rdb
4. Server and/or Client Certificate: these are file(s) with extensions .arm, .p12, .cer, .der, etc.
5. CA Signer Certificate, for example: Verisign, Thwarte, Entrust, internalCA, etc.

LDAP authentication related issues

There are many different cases of LDAP authentication issues.

For debugging, you will need all the files from “General configuration” on page 355 in addition to the following information.

LDAP connection (non-SSL)

1. Stop IHS.
2. Clear all logs in the /<IHS HOME>/logs directory.
3. Edit the httpd.conf; change LogLevel to debug.
4. Enable LDAP tracing:
 - Windows: Create a system variable called: LDAP_TRACE_FILE and set the value with the name for the log file, for example: c:\ldaptrace.log.
Create a system variable called: LDAP_DEBUG, and set the value to 65535.
 - UNIX: As the user ID that starts the IBM HTTP Server, create an environment variable called: LDAP_TRACE_FILE, and set the value with the name for the log file.

As the user ID that starts the IBM HTTP Server, create an environment variable called: LDAP_DEBUG and set the value to 65535.
5. Start IHS.
6. Recreate the problem.
7. Capture the activity with **netstat -na > netstat.out**.
8. Provide the following datafiles for review.
 - a. httpd.conf, error_log, access_log
 - b. netstat.out

- c. ldaptrace_log
- d. ldap.prop
- e. IHS version, LDAP Client version

Also include the date/time of failure along with the browser version and full URL (for example: <http://www.mycompany.com/mystuff/goodies/index.html>) that resulted in the LDAP failure.

LDAP connection over SSL

1. Stop IHS.
2. Clear all logs in the /<IHS HOME>/logs directory.
3. Edit the httpd.conf and change LogLevel to debug. Add the SSLTrace directive to the bottom of the httpd.conf file.
4. Enable LDAP tracing:
 - Windows: Create a system variable called: LDAP_TRACE_FILE, set the value with the name for the log file. Create a system variable called LDAP_DEBUG and set the value to 65535.
 - UNIX: As the user ID that starts the IBM HTTP Server, create an environment variable called LDAP_TRACE_FILE, set the value with the name for the log file.

As the user ID that starts the IBM HTTP Server, create an environment variable called LDAP_DEBUG, and set the value to 65535.
5. Enable GSKIT trace:
 - Windows: Create a system variable called GSK_TRACE_FILE and set the value with the name for the log file, for example: c:\gsktrace.log.
 - UNIX: As the user ID that starts the IBM HTTP Server, create an environment variable called GSK_TRACE_FILE and set the value with the name for the log file, for example: /usr/HTTPServer/logs/gsktrace_log.
6. Start IHS.
7. Recreate the problem.
8. Capture the activity with **netstat -na > netstat.out**.
9. Provide the following datafiles for review:
 - a. httpd.conf, error_log, access_log
 - b. netstat.out
 - c. ldaptrace_log
 - d. gsktrace_log
 - e. ldap.prop

- f. IHS version, LDAP Client version, Gskit version

Also include the date/time of failure along with the browser version and full URL (for example: <https://www.mycompany.com/mystuff/goodies/index.html>) that resulted in the LDAP failure.

Server hang issues

For debugging, you will need all the files from “General configuration” on page 355 in addition to the following information.

1. Stop IHS.
2. Edit the httpd.conf.
 - a. Change LogLevel to debug.
 - b. Enable Server-Status as described in the following document:
http://www-1.ibm.com/support/docview.wss?rs=180&context=SSEQTP&q=1008489&uid=swg21008489&loc=en_US&cs=utf-8&lang=en+en
3. Start IHS.
4. Capture the following data at the next occurrence of the HANG.
 - a. Issue the following command: **netstat -na >netstat1.out.**
 - b. Open the Web page in the browser: <http://<hostname>/server-status>.
 - c. Save the server-status output to serverstatus1.html; from the menu, select **File -> Save page as.**
 - d. Issue the following command: **ps -ef | grep httpd >httpdps1.out.**
 - e. Capture a stacktrace of the HUNG child process.
 - AIX:

```
dbx -a <pid>
(dbx) where >dbx1.out
(dbx) detach
```

This will allow you to exit dbx without killing the process.
 - Solaris:

```
pstack pid >pstack1.out
```
 - f. Wait three minutes.
 - g. Issue the command: **netstat -na >netstat2.out.**
 - h. Open this Web page in your browser: <http://hostname/server-status>.
 - i. Save the server-status output to serverstatus2.html.
 - j. Issue the command: **ps -ef | grep httpd >httpdps2.out**

k. Capture a stacktrace of the HUNG child process.

- AIX:

```
dbx -a <pid>
(dbx) where >dbx2.out
(dbx) detach
```

This will allow you to exit dbx without killing the process.

- Solaris:

```
pstack pid >pstack2.out
```

l. Wait three minutes.

m. Issue the command: **netstat -na >netstat3.out**.

n. In the browser, open the Web page: <http://hostname/server-status>.

o. Save the server-status output to serverstatus3.html.

p. Issue the command: **ps -ef |grep httpd >httpdps3.out**.

q. Capture a stacktrace of the HUNG child process:

- AIX:

```
dbx -a <pid>
(dbx) where >dbx3.out
(dbx) detach
```

This will allow you to exit dbx without killing the process.

- Solaris:

```
pstack pid >pstack3.out
```

- Windows:

If experiencing a HANG on a Windows platform, perform the **netstat** and **server-status** steps above, along with capturing one threaddump as described below:

```
drwtsn32 -p <process_id>
```

Note: You will not be able to request the /server-status output if "ALL" the httpd children processes are hung or busy. Therefore, skip the /server-status step if the page is not accessible at the time of the problem. This applies to both Windows and Unix operating systems.

5. Provide the following datafiles for review.

- a. httpd.conf
- b. error_log
- c. access_log

- d. serverstatus outputs
- e. dbx or pstack outputs on Unix. drwtsn32.log, user.dmp on Windows.
- f. netstat outputs
- g. ps outputs

Also include the date/time of the HANG and the IBM HTTP Server version information.

Back-end issues with the Application Server can cause the IBM HTTP Server to hang in the WebSphere Plugin.

Thread dumps of the JVM at the time of an IBM HTTP Server hang can also provide additional information. Therefore, please review the following technotes for more information on capturing JVM threaddumps during HANG conditions.

- ▶ JVM hang or performance degradation on AIX
<http://www-1.ibm.com/support/docview.wss?uid=swg21052641>
- ▶ JVM hang or performance degradation on Linux
<http://www-1.ibm.com/support/docview.wss?uid=swg21115785>
- ▶ JVM hang or performance degradation on HP-UX
<http://www-1.ibm.com/support/docview.wss?uid=swg21127574>
- ▶ JVM hang or performance degradation on Sun Solaris
<http://www-1.ibm.com/support/docview.wss?uid=swg21052644>
- ▶ JVM hang or performance degradation on Windows
<http://www-1.ibm.com/support/docview.wss?uid=swg21052640>
- ▶ JVM hang or performance degradation on Windows (5.0.X)
<http://www-1.ibm.com/support/docview.wss?uid=swg21111364>

Request failure (500, 404, 400, etc.) issues

For debugging, you will need all the files from “General configuration” on page 355 in addition to the following information.

1. Stop IHS.
2. Clear all logs in the <IHS_root>/logs directory.
3. Edit the httpd.conf and change Loglevel to debug.
4. Restart IHS.
5. Recreate the failure.
6. Provide the following datafiles for review.

- a. httpd.conf, error_log, access_log
- b. IBM HTTP Server version

Also include the date/time of failure along with the browser version and full URL (for example: http://www.mycompany.com/mystuff/goodies/index.html) that resulted in failure.

Dynamic content, tracing IBM HTTP Server and WebSphere connection

For debugging you will need all the files from “General configuration” on page 355 in addition to the following information.

1. Stop IHS and WebSphere.
2. Clear all logs in the <IHS_root>/logs directory.
3. Clear all logs in the <WebSphere_root>/logs directory.
4. Edit the plugin-cfg.xml and change Loglevel to Trace, for example: <Log LogLevel="Trace" Name="c:\WebSphere\AppServer\logs\native.log"/>
5. Edit httpd.conf and change Loglevel to debug.
6. On the application server, set the following trace string:
Servlet_Engine=all=enabled
 - a. In the Administrative Console, expand the Troubleshooting section and click **Logs and Trace**.
 - b. Click the link for your server.
 - c. Click **Diagnostic Trace**.
 - d. In the Trace Specification field, enter the following trace string:
Servlet_Engine=all=enabled
 - e. For Trace Output, choose the **File** radio button and enter a file name.
 - f. Choose the **Apply** button and save your configuration.
7. Restart IHS and WebSphere Application Server.
8. Recreate the failure.
9. Provide the following datafiles for review.
 - a. httpd.conf, error_log, access_log
 - b. plugin-cfg.xml
 - c. http_plugin.log
 - d. stderr and stdout

e. TraceFile from the previous steps.

f. IBM HTTP Server version

Also include the date/time of failure along with the browser version and full URL (for example: <http://www.mycompany.com/mystuff/goodies/index.jsp>) that resulted in failure.



B

Methodology

This appendix describes the methodology used in this book to document the key problem determination areas.

It is important to emphasize that this methodology is not a process for performing problem determination; it will not help to identify problems and solve them. It is a methodology to document problem determination. This methodology helps us to structure the book in such a way that it can be of more help to readers.

The methodology used in this book

First, you have to understand that the methodology introduced here is not a problem determination methodology, it is a methodology to produce problem determination guides.

Why do we need a methodology?

There are two main reasons why we need a methodology for documenting problems for a problem determination book:

1. It helps to organize the book and to document the problems in an organized manner for the book. The methodology can be used later so other books have a similar structure.
2. It helps the reader to follow the book and find the necessary information in the book.

How does the methodology work?

The following diagram is a visual representation of the methodology we used for this book.

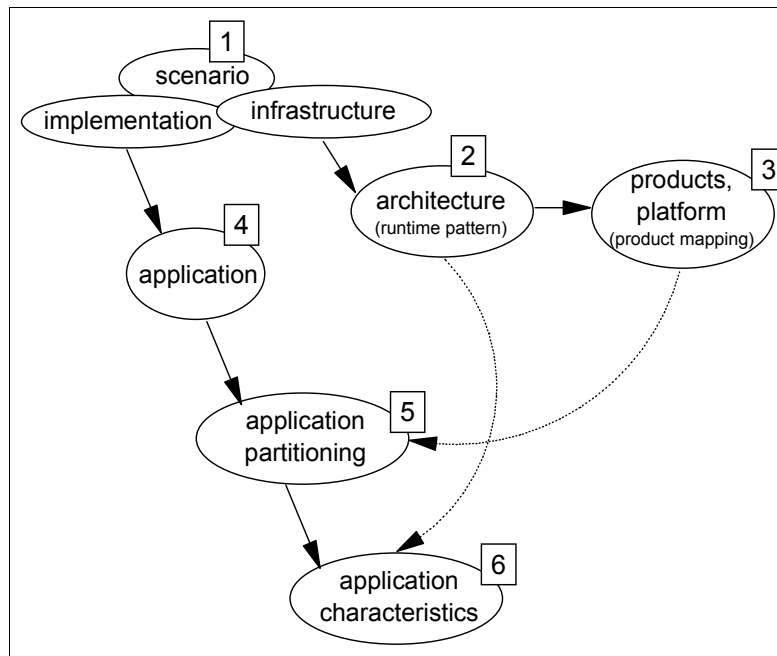


Figure B-1 Methodology steps

The following steps describe the diagram flow.

1. The first step is to identify the solution or scenario where problem determination is going to be performed. The scenario has a particular implementation and infrastructure.
2. This step is closely related to the Patterns for e-business methodology. We use the results of the Patterns for e-business methodology to determine further details of the solution infrastructure. The solution architecture is based on the Runtime patterns of the Patterns for e-business. It discovers the logical components and the characteristics of the architecture.

The architecture also determines the stage of the solution lifecycle. If the stage is development-test, the architecture is simply a development environment.

3. Once we have the details of the architecture, we can identify the products for the solution. This part relates to the Product mapping of the Patterns for e-business.
4. Describing the application is the first step towards the implementation. An application can be described in several ways, including use case diagrams, detailed descriptions of the functions, technical walkthroughs, etc. This description is more on the business requirement level than on the technical level.
5. The application implemented can be large and can consist of many parts. Partitioning the application is the first step towards revealing the technical details of the application.
6. After breaking up the application into smaller functional parts, we can start to identify the functional elements of the application, identify the components and describe their behavior. It is a low-level technical analysis of the application to discover the key characteristics.

There is a relation between the architecture and the application characteristics. The architecture can determine some of the application behavior and the configuration.

There is another relation between product mapping and application partitioning. Products determine some of the components on an application level, at least how functions are implemented.

Work products

The following diagram is a visual representation of how the methodology is applied to this book by instantiating a particular scenario.

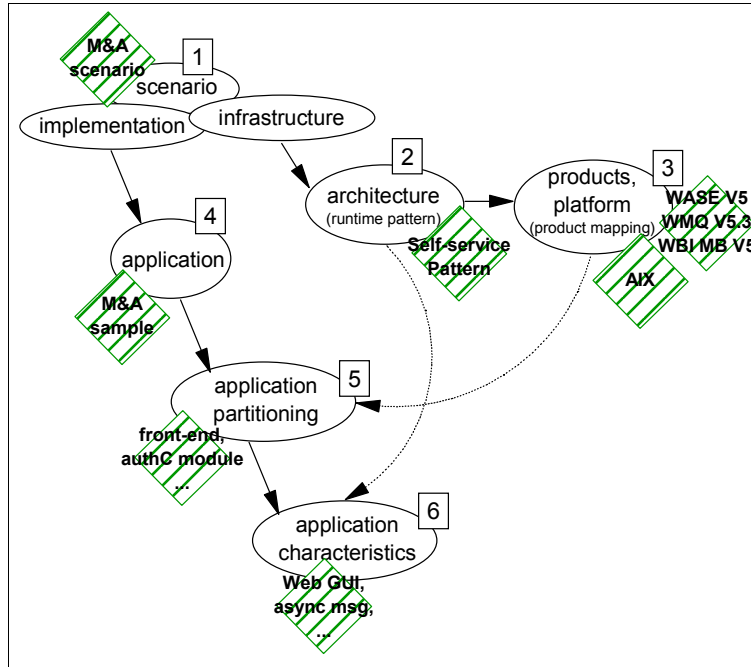


Figure B-2 Deliverables

The deliverables at each step are instances of a real-life solution implementation. The following is a high-level list covering aspects of the instantiated solution that the book is following.

1. The chosen scenario is called the eMerger scenario; for more information about the scenario, refer to 3.1, “The eMerge business scenario” on page 29.
2. The architecture is based on the Self-service pattern, defined in the Patterns for e-business. For more information about this pattern, refer to 3.4, “The Patterns for e-business approach” on page 35.
3. The chosen platform for this particular book is AIX, although there is not one homogenous platform in any real-life solution; this book will divert from the platform at times and provide some information about related platforms.

The products covered in this book are:

- WebSphere Application Server Enterprise
- WebSphere MQ
- WebSphere Business Integrator Message Broker

As with the platform, there are several other products involved in a real-life scenario such as eMerger. Although other products are not the focus of this book, we provide some information about related products and components.

4. The eMerge scenario also has a sample application. In this book, it is a collection of self-contained codes and components. For more information about the sample application, refer to 3.3, “Application details” on page 32.
5. High-level and detailed descriptions of the application functions are also available for the sample.
6. A more detailed description of the sample is also available, in order to identify the runtime components.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 374. Note that some of the documents referenced here may be available in softcopy only.

- ▶ *Self-Service Applications using IBM WebSphere V5.0 and IBM MQSeries Integrator*, SG24-6875
- ▶ *WebSphere Studio Application Developer Version 5 Programming Guide*, SG24-6957
- ▶ *IBM WebSphere V5.0 Performance, Scalability, and High Availability: WebSphere Handbook Series*, SG24-6192
- ▶ *WebSphere MQ for z/OS Messages and Codes*, GC33-0819
- ▶ *SecureWay Security Server LDAP Server Administration and Usage Guide*, SC24-5923

Online resources

These Web sites and URLs are also relevant as further information sources:

- ▶ IBM end of service (EOD) dates Web site:
<http://www-1.ibm.com/services/sl/swm/>
- ▶ IBM Software Support Web site
<http://www.ibm.com/software/support>
- ▶ IBM Support
<http://www.ibm.com/support>
- ▶ Voice access phone support number list
<http://techsupport.services.ibm.com/guides/contacts.html>
- ▶ Patterns for e-business Web page
<http://www.ibm.com/developerWorks/patterns/>

- ▶ IBM HTTP Server problem determination
<http://www-306.ibm.com/software/webservers/httpservers/support/>
- ▶ WebSphere Application Server Enterprise problem determination
<http://www-306.ibm.com/software/webservers/appserv/was/support/>
- ▶ WebSphere MQ problem determination
<http://www-306.ibm.com/software/integration/wmq/support/>
- ▶ WebSphere Business Integration Message Broker problem determination
<http://www-306.ibm.com/software/integration/wbmessagebroker/support/>
- ▶ IBM DB2 problem determination
<http://www-306.ibm.com/software/data/db2/udb/support.html>
<http://www-106.ibm.com/developerworks/db2/library/techarticle/0205bargas/0205bargas.html>
- ▶ Google Web site
<http://www.google.com>
- ▶ JVM diagnostics
<http://www-106.ibm.com/developerworks/java/jdk/diagnosis/>
- ▶ WebSphere Studio Application Developer Log files
http://www-106.ibm.com/developerworks/websphere/library/techarticles/0301_cartledge/cartledge.html
- ▶ AlphaWorks Log and Trace Analyzer
<http://www.user ID.ibm.com/tech/logandtrace>
- ▶ J2EE code validation tool
http://www-106.ibm.com/developerworks/websphere/downloads/j2ee_code_validation.html
- ▶ RFH util
<http://www-3.ibm.com/software/integration/support/supportpacs/individual/ih03.html>
- ▶ DeveloperWorks - WebSphere
<http://www.ibm.com/developerworks/websphere/>
- ▶ IBM JVM diagnosis
<http://www.ibm.com/developerworks/java/jdk/diagnosis/>
- ▶ DB2 problem determination
<http://www-106.ibm.com/developerworks/db2/library/techarticle/0204vangennip/0204vangennip1.html>

- ▶ Extended Messaging Formatter and Parser article
http://www-106.ibm.com/developerworks/websphere/techjournal/0401_green/green.html
- ▶ WebSphere InfoCenter
<http://publib.boulder.ibm.com/infocenter/wasinfo/index.jsp>
- ▶ TechNote: *The WebSphere Application Server 5.0 Embedded JMS Broker cannot be started when Embedded JMS is configure to run with Non-root User ID*
http://www-1.ibm.com/support/docview.wss?rs=180&context=SSEQTP&q1=jmsserver&uid=swg21144021&loc=en_US&cs=utf-8&lang=en
- ▶ IBM HTTP Server article
http://www-1.ibm.com/support/docview.wss?rs=180&context=SSEQTP&q=1008489&uid=swg21008489&loc=en_US&cs=utf-8&lang=en+en
- ▶ JVM hang or performance degradation on AIX
<http://www-1.ibm.com/support/docview.wss?uid=swg21052641>
- ▶ JVM hang or performance degradation on Linux
<http://www-1.ibm.com/support/docview.wss?uid=swg21115785>
- ▶ JVM hang or performance degradation on HP-UX
<http://www-1.ibm.com/support/docview.wss?uid=swg21127574>
- ▶ JVM hang or performance degradation on Sun Solaris
<http://www-1.ibm.com/support/docview.wss?uid=swg21052644>
- ▶ JVM hang or performance degradation on Windows
<http://www-1.ibm.com/support/docview.wss?uid=swg21052640>
- ▶ JVM hang or performance degradation on Windows (5.0.X)
<http://www-1.ibm.com/support/docview.wss?uid=swg21111364>

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, TechNotes, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

- ▶ IBM Support and downloads

ibm.com/support

- ▶ IBM Global Services

ibm.com/services

Index

A

- AbstractMethodError 155
- AccessControlExceptions 167
- active participation 23
- Active Script debugging 117
- ActivitySession service 207
 - NotSupportedException 207
 - Things to check 207
- additional information 23
- addNode
 - trace 219
- addNode.log 64, 218
- AdminConfig 142
- adminconsole 141
- Administrative Console
 - JVM logs 53
 - permissions 141
 - save conflicts 141
 - save fails 142
 - Things to check 141
 - trace 142
- AIX 44
- alphaWorks 113
- analysis steps 18
- Analyzing the problem 19
- anomaly 20
- Apache Web server 128
- APAR 20
- Application Assembly 102
- Application database 146
- Application flow 139
- Application pattern 35
- Application profiling 202
 - Things to check 203
 - Unexpected concurrency 203
- Application Server Toolkit 59
- Application servers 137
- Application start 146, 167
- Application unavailable 142
- applications
 - lost 223
 - Things to check 145
- Asynchronous Beans 197

- Things to check 197
- asynchronous request 35
- asynchronous scope objects 197
- Attach to Process Engine 110
- Authentication fails 164
- Authorization failed 148
- authorized program analysis report 20
- Autonomic Computing 113
- Auxiliary Trace facility 304

B

- back-end applications 39
- back-end systems 37
- background information 21
- Backup cluster support 207
 - application failure 208
 - failover 208
 - Fallback 209
 - Things to check 208
- BackupConfig 65
- binary trace file
 - formatting 311
- bootstrap port 151
- BOOTSTRAP_ADDRESS 169
- BPE 174
- BPEAuditLogDelete 174
- BPEIntQueue 184
- BPESystemAdministrator role 181
- BRBeans.jar 191
- Browser errors 148
- business process container
 - troubleshoot 174
- business processes 14
- Business rule beans 190
 - Database deadlock 191
 - Rules trigger 192
 - Things to check 191
 - trace 191
- business scenario 29
- BusinessRuleBeans exception 191

C

- Cannot login 148

- CannotInstantiateObjectException 150, 253
- capturing documentation 23
- CCSID 189
- CEEDUMP 334
- Cell 212
- CETR 321
- CICS 302
- CICS Application Server
 - processes 305
- CICS auxiliary trace 321
- CICS internal trace 320
- CICS messages 319
- CICS processes dump 307
- CICS region 303
- CICS Trace Control Facility 321
- CICS Transaction Gateway 307
 - Application Trace 315
 - daemon 314
 - daemon trace 316
 - EXCI Trace 318
 - JNI Tracing 317
 - Trace 313
 - z/OS 313
- CICS Transaction GatewayTrace file 314
- CICS Universal Client 307
 - default trace files 309
 - trace 309
- CICSCLI 308
- CICSFTRC 309, 311
- cicsservice 304
- CICSTRACE 304
- classes directory 147
- ClassNotFoundException 177
- Client daemon 308
- Client daemon trace 309
 - analysis 312
 - wrapping 311
- Clients
 - Things to check 150
 - WebSphere Application Server 150
- Closing problem 20
- Cloudscape 175
- Cluster 212
- collector 59
- Collector Tool
 - WebSphere Application Server 58
- CommTrace sample 256
- CommunicationException 151
- company merger 29
- compensation 176
- Compiled language debugger 59
- Components 225
- concurrent transaction 186
- ConnectionWaitTimeoutException 157
- Console groups 168
- console log 50
- Console users 168
- Container Managed Persistence 149
- contract 12
- core dump analysis 45
- Coupling Facility 341
- CPU usage 157
- CPU utilization 157
- critical business impact 22
- critical components 11
- CSMT 302
- CTG_JNI_TRACE 313
- ctgstart 313–314
- CTIBBOxx PARMLIB member 343
- CTRACE
 - address space 345
- current.env 343
- Custom login page 164

D

- Data sources 155
 - Deadlock 157
 - Things to check 156
- database trace 153
- datasource 154
- DB2 Custom properties 154
- db2java.zip 177
- DB2sqljtrace 97
- db2support 96
- dbx 45
 - sample 46
- DCE
 - Endpoint mapper 303
 - messages 303
 - Security messages 303
- dead-letter queue 70
- deadlock 139, 175, 185
- Deployment Manager 212
 - base servers 213
 - Enterprise servers 214
- Deployment manager
 - addNode 217

- ADMC0016E 222
- Administrative Console 223
- ADMN0022E 224
- ADMU0124E 217
- applications lost 223
- Domain Name Server 220
- process status 224
- Things to check 223
- Workload Management 225
- Distributed sessions 238
- DNS 143
- Documentation 13–14
- documentation
 - error messages 13
- Documenting
 - problem determination 14
 - the problem 18
- domain name 143
- Domino Web server 128
- dspmqcsv 266
- DtraceSettingsFile 63
- dumpNameSpace 65, 168
- dumpnamespace 157
- dumpthread.jacl 139
- Dynamic query 198
 - Query fails 199
 - Things to check 198

E

- EE Test Environment 104
- EID 336
- EJB 149
- EJB module fails 149
- EJB request
 - distributed 229
- embedded HTTP Server 128
- Embedded Messaging 159
 - install problem 159
 - Things to check 159
- eMerge business scenario 29
- Emergency situation 16
- Enable core dump 48
- Encina
 - trace 305
 - trace messages 303
- end of service date 12
- enterprise application install 188
- EOD 12

- Error
 - 403 148
- error log 50
- errpt 50
 - sample 50
- escalation process 15
- ESQL debug 92
- event identifiers 336
- Extended messaging 192
 - receive message 194
 - send message 196
 - Things to check 193
- External
 - resources 11
 - support 11
- External fault 20
- external vendor 12
- External Web services 37

F

- FFST 67
 - Function Stack 67
 - Trace History 67
- finally{} block 139
- firewall 141
- First Failure Data Capture tool 64
- first symptom 17
- First-Failure Support Technology 67
- Flow Debug 90
 - perspective 91
 - view 91
- functional problems 5

G

- Gateway daemon 313
- gdb 45
- General problem determination 136
- generalized trace facility 335
- Generic JMS providers 158
- Get Customer Details use case 32
- ghost process 137
- GIOP Request 258
- global namespace 253
- gsk library 132
- GSKit 355
- GTF 335

H

- Hardware fault 20
- HFS 96
- highest severity 22
- Host Aliases 138
- hosts file 143
- Hot Method replace 116
- HTTP Error
 - 404 132
 - 500 129
- HTTP request 227
- HTTP session
 - Database-based persistence 241
 - Memory-to-memory replication 243
- HTTP Session Management 236
 - Lost sessions 239
 - Persistent HTTP Sessions 241
 - Things to Check 238
 - trace 249
- HTTP Session Manager
 - trace 249
- HTTP Sessions 171
- http_plugin.log 65, 127

I

- IBM Agent Controller 103
- IBM DB2 UDB 94
 - db2diag.log 95
 - db2support 95
 - Dump files 95
 - JDBC trace 96
 - Messages files 95
 - Trap files 95
- IBM Global Services 24
- IBM HTTP Server 94, 128, 355
 - access.log 94
 - CMS key database 357
 - Dynamic Content 363
 - error.log 94
 - General configuration 355
 - Global Security Kit 355
 - HTTP Request Failures 362
 - LDAP authentication 358
 - Server hang issues 360
 - SSL Handshake 356
- IBM products 4
- IBM ServiceLink 25
- IBM Support Center 18

- ibmslapd 98
- Identifying the problem 17
- IGS 24
- IllegalConnectionUseException 157
- Implementing problem resolution 19
- Include Applications 223
- initial resolution 18
- integration middleware 39
- Interactive Problem Control Facility 345
- Interim Fix 20
- Internal Server Error 129
- Internationalization service 205
 - JNI lookup 205
 - Things to check 205
- Internet Access 24
- Internet Information Services Web server 129
- InvalidExecutableException 250
- IPCS 345
- iPlanet Web server 129
- iptrace 45

J

- J2EE Applications 144
- J2EE client hangs 152
- J2EE Code Validation 118
- J9 JVM 116
- java
 - comp/env 253
- Java 2 security
 - errors 167
 - Things to check 167
- Java Database Connector Tracing 96
- Java development tools debugger 59
- Java heap memory 139
- Java snippets 185
- Java stack trace 45
- Java Virtual Machine 52
 - Memory 169
- Java2 security 146
- JavaScript 117
 - debug adapter 59
- JDBC Provider 154, 249
- JDBC providers 152
 - Things to check 152
- JDT debugger 59
- JMS Provider 250
 - Things to check 250
- JMS provider 193

- JMS resources 158
 - Message Prefixes 162
 - not found 161
 - Things to check 158
- JMS Server
 - start 161
- jmsserver 251
 - start 250
- JNDI 157, 252
- JNDI client 252
- JNDI names 193
- JNDI namespace 104
- JNDI Naming 150, 168
 - Things to check 168
- Joblog 352
- JOBNAMEP option 336
- JRAS 346
- JRas tracing 145
- JSP test 139
- JVM 52
 - Diagnostics Guide 141
 - heap size 139
 - Hot Method replace 116
- JVM diagnostic 52
- JVM logs 52
 - Administrative Console 53
 - ClassName 55
 - Component 55
 - EventType 55
 - interpreting 54
 - LongName 55
 - Message formats 54
 - MethodName 55
 - Organization 55
 - Product 55
 - review 53
 - ShortName 54
 - ThreadId 54
 - TimeStamp 54
- JVM Memory
 - Things to check 169

K

- Keyring 133

L

- Last participant support 205
 - heuristic condition 206

- Things to check 206
- Transaction roll back 206
- WTRN0061W 206
- launchClient 151, 235
 - hangs 151
- LDAP 97, 343
 - z/OS 353
- LDAP debug categories 99
- ldtrc 98
- Lessons learned 15
- lib_security 132
- Lightweight Directory Access Protocol 97
- listener bind 161
- listener ports 193
- listi 46
- Load Balancer 225
- Log Analyzer 53, 56
 - analyzing logs 111
 - service log 56
- Log Analyzer tool 114
- log files 19
- Login error 148, 164
- Login page loops 143
- loopback address 220
- lowest severity 22
- lspp 45
- LTPA 141, 163
- LTPAServerObject 165

M

- Managed Process 212
- MAXMSGSL 268
- MBean
 - access denied 224
- meet-in-the-middle mapping 199
- memory leak 139
- Message Broker 282
 - Agent Controller 90
 - attach workbench debugger 289
 - authorities 293
 - BIP0889E 293
 - BIP1511E 292
 - BIP2066E 290
 - BIP8053E 290
 - BIP8075E 292
 - Breakpoint 91
 - broker archive 298
 - Broker delete 289

- broker delete 295
- broker execution group trace 81
- broker recreate 292
- Broker start 287–288
- Broker start (z/OS) 294
- broker stop 295
- check list 80
- com.ibm.broker.plugin 291
- Compute node 290
- ConfigManagerProxy 291
- Configuration Manager start 295
- correlation name error 299
- create Configuration Manager 290
- database 285
- database access 288
- Debugger 90
- deploy flows 290
- deploying 296
- deployment 299
- environment variables 283
- ESQL 296
- ESQL debug 92
- Execution group 297
- Getting Started wizard 294
- input queue 297
- message flow 283
- message flow deploy 298
- message references 297
- ODBC Trace 85
- ODBC trace 80
- rename flow 298
- Run To Return 93
- Service Trace 85
- service updates 286
- Step Into 93
- Step Into Source Code 92
- Step Over 93
- task list 298
- Things to check 282
- Toolkit 300
- trace node 86
 - example 87
- Tracing 80
- User Trace 82
- Workspace 300
- Message Brokers Toolkit 90
- Message Oriented Middleware 39
- Message Queue Interface 264
- Messaging provider 158
- methodology 366
- minimal business impact 22
- Model-View-Controller 35
- Monitoring Level 248
- mq_install.log 159
- MQFB_APPL_CANNOT_BE_STARTED 278
- MQI 264
- MQJMS_LIB_ROOT 158, 182
- mqm group 158
- MQRC_CLUSTER_PUT_INHIBITED 274
- MQRC_NO_MSG_AVAILABLE 269
- MQRC_UNKNOWN_OBJECT_NAME 275
- mqsichangebroker 287
- mqsichangetrace 81
- mqsicreateaclgroup 293
- mqsdeletebroker 296
- mqsiformatlog 81
- mqsireadlog 81
- mqsistart 81
- mqsistop 81, 295
- MSGAMQ9555 277
- multiple problems 17
- multiple threads 140
- MVC 35

N

- Name not found 169
- named queue 70
- NameNotFoundException 149, 181, 254
- Naming 168
 - Qualified names 252
 - Relative names 251
 - Things to check 252
- national characters 175
- Native Library Path 158
- Netscape browser 141, 150
- Network Deployment 212
 - Naming 251
 - Resources 249
- never ending transactions 147
- Node 212
- Node Federation 215
- Nodes
 - synchronize 224
- NoRuleFoundException 191
- NullPointerException 148
- numIncomingNonWLMObjectRequests 230
- numIncomingRequests 230

numIncomingStrongAffinityRequests 230

O

Object pools 203
 deadlock 204
 object reuse 204
 Synchronization 204
 Things to check 203
Object Request Broker 228
Operating System 44
 Monitoring a running process 49
 process 45
 release 44
 segmentation fault 46
 Tracing 45
 version 44
OperationNotSupportedException 254
ORB 228, 255
 Fragmentation 260
 Identifying failure 255
 Marshal Exception 260
 Trace 256
Organizing problem resolution 18
oslevel 45

P

Patterns for e-business approach 35
PCF command 265
People and skills 11
Performance 171
Performance Monitoring Service 66
pid file 65
plug-in 38, 124
 configuration 130
 log file 127
plugin-cfg.xml 124, 227
PMEinstallSummary.log 65
port conflict 138
previous knowledge 14
PrimaryServers list 227
problem
 escalation 15
problem analysis 19
 skills 19
Problem context 14
problem definition 21
Problem description 14
Problem determination 5, 10

 fundamentals 16
 steps 16
problem documentation 20
problem escalation 12
problem identification 17
 questions 17
problem owner 10, 14
problem ownership 10, 18
problem prevention 10
problem reproduction 13
problem resolution
 basic questions 21
 closing 20
 implementation 19
 organizing 18
problem severity 16, 22
Problem status 14
process application
 install 177
Process Choreographer 174
 Activity description 177
 audit trail 174
 BPEA0010E 175
 BPECContainer startup 181
 BPED0000E 188
 BPPE0031E 179
 BPEU0002E 181
 BPEU0024E 180
 container install 180
 database tables 174
 Deadlocks with compensation 176
 jms/BPECF not found 181
 non-interruptible process timeout 178
 Nullpointer Exception 184
 People Activity 186
 process start 179
 process template deployment 176
 Quiesce/Resume algorithm 184
 Resource reference 183
 Syncpoint Manager Exceptions 187
 Things to check 174
 trace 174
 Transport Type 189
 Variables access 185
 Web client 175, 178
 work items 182
process debugger
 Java debugger 109
 remote 110

- Process Definition 140
- Process logs 55
- process of elimination 5
- product documentation 13
- product support 12
- production environment 13
 - reproduce 13
- Products 38
- Program Temporary Fix 20
- PTF 20

Q

- Queue Connection Factories 158
- Queue Destinations 158
- Queue manager
 - start 160

R

- RAC 112
- RASStart 289
- recreate problem 15
- Redbooks Web site 374
 - Contact us xvi
- Reference object 254
- regedit.exe 328
- registers 46
- relevant diagnostic information 22
- relevant manuals 19
- remote queue managers
 - configure 75
- removeNode 222, 225
- removeNode.log 64
- Request and Accept Quote use case 33
- request process 124
- Resource providers 152
- Resource Recovery Services 349
- responsibilities 24
- RestoreConfig 65
- retention queue 184
- RFHUTIL 119
- RRS 349
- Runtime pattern 36

S

- Sample application
 - products 38
- sample application 29

- use cases 32
- sample architecture 30
- Scenario
 - products 38
- scenario architecture 30
- Schedule task 201
- Scheduler 200
 - polling interval 200
 - scheduled task 201
 - task timing 202
 - Things to check 200
- SDSF 352
- Secure Socket Layer 163
- Security 138, 141
 - disable 163
 - Java 2 security 167
 - SSL 165
 - Things to check 162
 - WebSphere Application Server 162
- security permissions 146, 158
- security roles 191
- security settings 163
- Self Service
 - Decomposition pattern 35
- server process ID 65
- serverStatus.log 64
- service log 53, 56
 - Log Analyzer 56
- ServiceUnavailableException 151
- servlet engine threads 139
- Session affinity 130
- Session management
 - inheritance 236
- Session Manager 130
- Session monitor 245
- Session scope 237
- session time out 141
- Session tracking
 - cookies 236
 - SSL information 236
 - URL rewriting 236
- Severity 1 22
- Severity 2 22
- Severity 3 22
- Severity 4 22
- Shared httpsession context 237
- showlog script 56
- SID 295
- significant business impact 22

- Single Point of Contact 25
- Single Sign On 163
- smitty chgsys 48
- SMUI 332
 - Communications trace 334
 - Message log 333
 - Trace writer 334
- SOAP connector 216
- Software configuration error 19
- Software installation error 19
- Software Maintenance offering 25
- software problem report 23
- Software product fault 20
- Software Support 24
 - questions 24
- some business impact 22
- specific problem 17
- SPOC 25
- SQLJ debug adapter 117
- SQLJ debugging 117
- SSL 130, 132, 163
 - handshake 166
 - Things to check 165
 - token 165
- SSLHandshakeException 166
- SSO 163
- stack trace 148
- Staff resolution 182
- staff verb 183
- startServer.log 64
- Startup beans 202
 - Application start 202
 - Things to check 202
- static content 133
- static IP address 143
- stopServer.log 64
- strmqcsv 266
- Struts 178
- sub-processes 14
- subsystem failure 18
- Support Line offering 25
- Symptom Database 111
 - update 57
- SYMREC file 302
- synchronized environment 15
- synchronous request 35
- SYSBBOSS 343
- SYSLOG 352
- Syslog 352

- Syslog Daemon 51
 - system availability 18
 - system core dump 45
 - system dump 50
 - system log 50
- System Management User
 - Communications trace 332
 - Debug mode 333
 - Message log 332
 - Trace writer 332
- system recovery 18
- System trace records 307
- system-call tracing facility 45

T

- TCP/IP sprayer 225
- tcpdump 45
- Technical resources 12
- template already exists 176
- Test configuration 15
- Test Connection 156
- test environment 13
- Things to Check
 - HTTP Session Management 238
- Things to check 10, 126
 - ActivitySession service 207
 - Administrative Console 141
 - Application profiling 203
 - applications 145
 - Asynchronous Beans 197
 - Backup cluster support 208
 - Business rule beans 191
 - Clients 150
 - Data sources 156
 - Deployment manager 223
 - Dynamic query 198
 - Embedded Messaging 159
 - Extended messaging 193
 - Internationalization service 205
 - Java 2 security 167
 - JDBC providers 152
 - JMS Provider 250
 - JMS Resources 158
 - JNDI Naming 168
 - JVM Memory 169
 - Last participant support 206
 - Message Broker 282
 - Naming 252

- Object pools 203
- Process Choreographer 174
- Scheduler 200
- Security 162
- SSL 165
- Startup beans 202
- Tracing 149
- WebSphere Application Server 137
- WebSphere Enterprise 215
- WebSphere MQ 264
- WorkArea service 204
- Workload management 228
- wsadmin 143
- thread dump 139
- time-critical situations 11
- Tivoli Directory Server 97
- Tivoli Performance Viewer 65, 139, 245
- tkadmin 305
- Topic Connection Factories 158
- Topic Destinations 158
- trace 45
- trace a running server 62
- Trace file 149
 - create 150
- trace in-storage buffers 307
- trace levels 344
- Tracing
 - Things to check 149
- Tracing API 315
- Transaction time out 149
- transactions 147
- transient data queue 302
- trcrpt 45, 73
 - sample 74
- trcstop 45
- Tuning 5
- TXSeries
 - system tracing start 305
 - system tracing stop 305
 - trace 304
 - trace files 305

U

- ulimit 48
- UNAUTHENTICATED 148
- undetected problem 17
- Universal Test Client 104, 169
- UnsatisfiedLinkError 155

- URL rewriting 130
- User error 19
- user registry
 - LDAP 138
- UTC 104

V

- verboseGC 170
- Version Info
 - WebSphere Application Server 57
- versionInfo 57, 216
 - sample 57
- VirtualHost 138
- VisualBasic 117
- Voice access 25

W

- was.policy 146, 167
- WAS40DataSources 157
- waslogbr 56
- Web server 38
- Web Server plug-in 124
 - LogLevel 127
 - Requests processing algorithm 124
 - Things to check 126
- Web Server Redirector component 37
- WebSphere Application Server 38, 52, 136
 - Administrative Console 141
 - application client trace 63
 - applications 144
 - Clients 150
 - Collector Tool 58
 - debug adapter 59
 - First Failure Data Capture Tool 64
 - JMS trace 63
 - JVM logs 52
 - log file 136
 - Process logs 55
 - Security 162
 - security errors 138
 - server hangs 138
 - server startup trace 61
 - stand-alone process trace 63
 - stderr stream 52
 - stdout stream 52
 - SystemErr.log 53
 - SystemOut.log 53
 - Things to check 137

- trace a running server 62
- Trace strings 60
- Tracing 59, 149
- Version Info 57
- WebSphere MQ 265
- z/OS 340
- WebSphere Application Server Enterprise 52
- WebSphere Business Integration Message Broker 80, 282
 - Windows 330
- WebSphere Business Integrator
 - z/OS 347
- WebSphere Enterprise
 - Administrative Console 209
 - Business rule beans 190
 - Things to check 215
- WebSphere JMS Provider 158
- WebSphere MQ 66, 264
 - AMQ9697 279
 - Asynchronous trace 72, 75
 - Client 271
 - Commands 265
 - Configuration files 71
 - Dead Letter Handler 280
 - Dead-letter queues 70
 - Early errors 70
 - Error Logs 68
 - incorrect output 268
 - Interactive trace 72, 75
 - Java Tracing 73
 - Log files 69
 - MQJMS2013 280
 - Operator messages 70
 - queue 266
 - receive 265
 - remote queue 267
 - slow system 267
 - trace 72
 - trace formatting 73
 - Tracing 71
 - WebSphere Application Server 265
 - Windows 328
 - z/OS 334
- WebSphere MQ channels 276
 - SNA sender 278
 - socket files 277
 - start 276–278
- WebSphere MQ client
 - connection 272
 - terminating 271
- WebSphere MQ cluster 272
 - cluster queue 274
 - cluster resolution 275
 - Queue manager fails 274
 - queue managers 273
 - Repositories lost 275
 - Repository fails 274
- WebSphere MQ Explorer 75
 - Browse Messages 78
 - Channels 79
 - Choose columns 77
 - monitor 77
 - Queue manager 79
 - Refresh 77
 - review 77
 - single queue 77
- WebSphere MQ JMS Provider 158
- WebSphere MQ security 278
 - Changing permissions 279
 - client connection 279
 - Invalid authentication 280
 - SSL Channel 278
 - WebSphere MQ start 279
- WebSphere MQ Things to check 264
- WebSphere Studio 102
 - Administrative Console 104
 - Component test 103
 - Debugger 106
 - Local debug 106
 - Monitoring 111
 - Process debugger 107
 - Profiling perspective 110
 - Remote Agent Controller 112
 - Remote debug 106
 - TCP/IP Monitor 112
- WebSphere Studio Application Developer 113
- WebSphere Studio Application Developer Integration Edition 102
- WebSphere Studio Profiling tool 114
- WebSphere Test Environment 103
 - Logs and trace 105
- WebSphere TXSeries 302
 - trace 304
- Windows 326
 - Event Viewer 326
 - Java 328
 - Message Broker
 - Configmgr and Broker tracing 331

- ODBC trace 330
- Performance 327
- Registry Editor 328
- Task Manager 326
- WebSphere Business Integration Message Broker 330
- WebSphere MQ 328
 - Error Logs 329
 - Tracing 329
- Windows 2000 Server 326
- work manager 197
- WorkArea service 204
 - Remote invocations 204
 - Things to check 204
- Workload Management
 - HTTP Requests 225
- Workload management 131, 225
 - CORBA 233
 - data counters 234
 - EJB Requests 228
 - failover 231
 - PMI data 233
 - Servlet Requests 226
 - Things to check 228
 - workload share 231
 - WWLM0061W 232
 - WWLM0062W 232
 - WWLM0063W 232
 - WWLM0064W 232
 - WWLM0065W 233
- worst case scenario 16
- ws_transport 133
- wsadmin 143
 - hangs 144
 - SOAP connection 144
 - Things to check 143
- wsadmin.traceout 64

- Diagnostic Data 341
- Joblog 341
- SYSLOG 341
- WebSphere Business Integrator 347
 - address spaces 349
 - Administration Agent 348
 - Components 347
 - Control Process 348
 - DB2 349
 - Execution Group 348
 - execution group trace 350
 - HFS files 352
 - logs 352
 - OMVS 349
 - RRS 349
 - service trace 351
 - Trace files 353
 - trace status 350
 - User Name Server 348
 - User Process 348
 - WebSphere MQ 349
- WebSphere MQ 334
 - Channel trace 338

Z

- z/OS 332
 - CTRACE for WebSphere 343
 - IPCS and WebSphere MQ 339
 - Language Environment 334
 - LDAP 353
 - trace 354
 - System Management User Interface 332
 - WebSphere Application Server 340
 - CTRACE 343



Problem Determination Across Multiple WebSphere Products

(0.5" spine)
0.475" <-> 0.875"
250 <-> 459 pages



Problem Determination Across Multiple WebSphere Products AIX Platform

**WebSphere
Application Server
base, Enterprise,
Network Deployment**

This IBM Redbook provides detailed information about problem determination for various WebSphere products in a real-life scenario. It is a valuable resource of information for IT administrators, IT specialists and application developers.

**WebSphere MQ,
Business Integrator
Message Broker**

Part 1, “Introduction” offers an explanation of how the book is organized and how it should be used for problem determination. It also provides details about the solution chosen for the book to better understand the environment and to provide a real-life scenario.

**Addressing generic
runtime problems**

Part 2, “Problem Determination details” is the core of the whole book. This part starts with two chapters discussing problem determination tools, one chapter for the runtime environment (AIX) and one for the development environment (mainly WebSphere Studio Application Developer V5). The following chapters go into details about problems (symptoms) with each product: WebSphere Application Server Enterprise V5, WebSphere MQ V5.3 and WebSphere Business Integration Message Broker V5.

The “Appendixes” extend the book with problem determination tools on other platforms, mainly z/OS. It also provides insight into the methodology used for this book to document problems, tools and problem determination techniques.

**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**