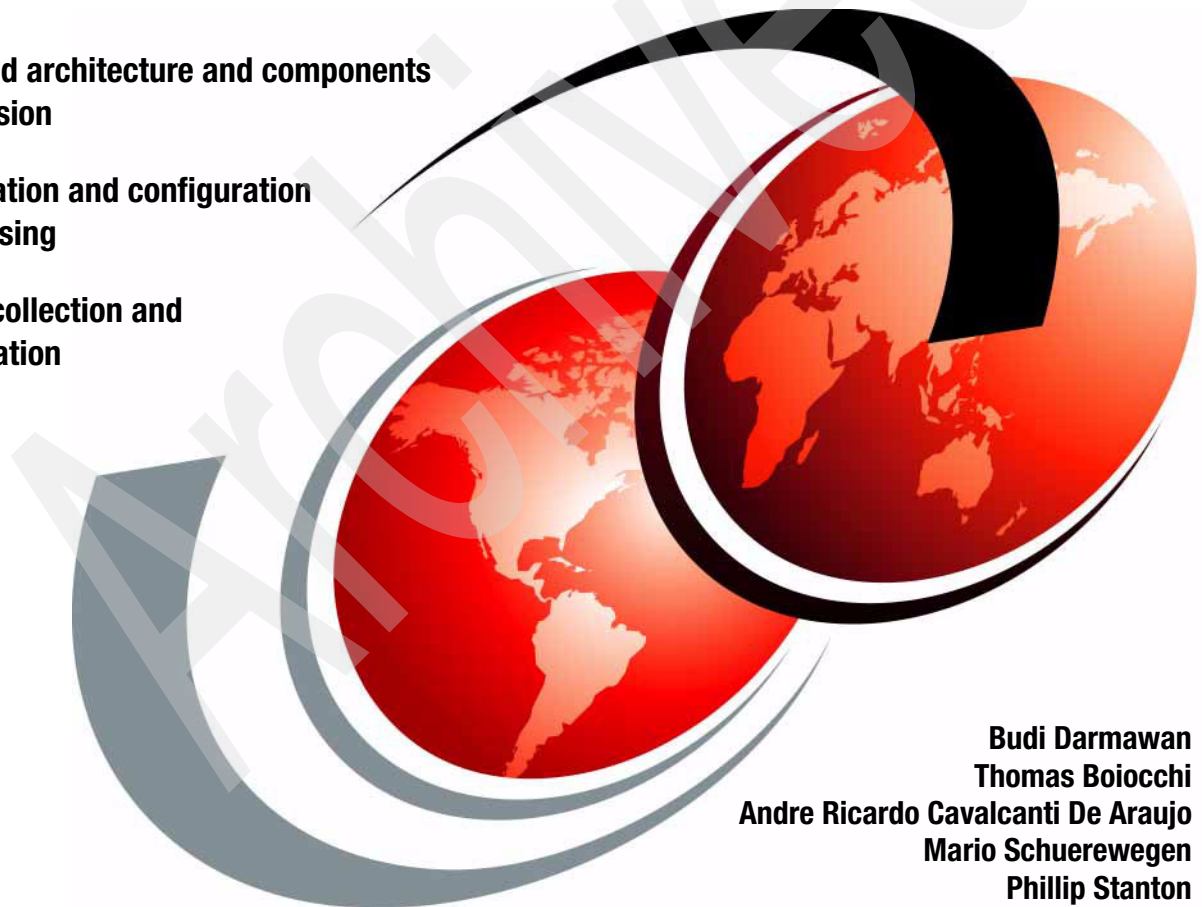


Certification Guide Series: IBM Tivoli Netcool/OMNibus V7.2 Implementation

Detailed architecture and components
discussion

Installation and configuration
processing

Event collection and
automation



Budi Darmawan
Thomas Boiocchi
Andre Ricardo Cavalcanti De Araujo
Mario Schuerewegen
Phillip Stanton



International Technical Support Organization

**Certification Guide Series: IBM Tivoli
Netcool/OMNibus V7.2 Implementation**

September 2009

Archived

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

First Edition (September 2009)

This edition applies to Version 7, Release 2, Modification 0 of IBM Tivoli Netcool/OMNibus (product number 5724-S44).

© Copyright International Business Machines Corporation 2009. 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

Figures	vii
Tables	xi
Examples	xiii
Notices	xv
Trademarks	xvi
Preface	xvii
The team who wrote this book	xvii
Become a published author	xix
Comments welcome	xx
Chapter 1. Certification overview	1
1.1 IBM Professional Certification Program	2
1.1.1 Benefits of certification	3
1.1.2 Tivoli Software Professional Certification	4
1.1.3 Growth through skills	6
1.2 IBM Tivoli Netcool/OMNIBus V7.2 test objectives	8
1.2.1 Planning	9
1.2.2 Installation	9
1.2.3 Configuration	11
1.2.4 Performance tuning and problem determination	23
1.2.5 Administration	25
1.2.6 Training	25
1.3 Certification achieved	26
1.3.1 Tivoli Netcool Core V3.0	26
1.3.2 Tivoli Netcool Impact V4.0	29
1.3.3 Tivoli Fault Management Solutions 2008	31
1.3.4 Tivoli Performance Management Solutions 2008	32
1.3.5 Tivoli Business Application Management Solutions 2008	32
1.3.6 IBM Service Management Network and Service Assurance 2009 ..	33
1.3.7 IBM Service Management Data Center Management and Transformation 2009	34
1.4 Recommended study resources	35
1.4.1 Classroom courses	35
1.4.2 Online course	36

Chapter 2. Planning	37
2.1 IBM Tivoli Netcool/OMNIbus components	38
2.1.1 Architectural components	38
2.1.2 Desktop and administrative components	40
2.1.3 Groups, roles, and users	41
2.1.4 Insert Delete Update Cycle (IDUC) components	44
2.2 IBM Tivoli Netcool/OMNIbus configuration	45
2.2.1 Without failover	45
2.2.2 With failover	46
2.2.3 Desktop ObjectServer	47
2.2.4 Tiered implementation	48
2.2.5 Probe features	49
2.3 Configuration planning issues	49
2.3.1 Naming convention	50
2.3.2 SSL usage	50
2.3.3 Port and firewall considerations	52
2.4 ObjectServer tables	52
Chapter 3. Installation	61
3.1 Installation process	62
3.2 Simnet probe installation	65
3.3 Installing a FixPack	66
3.4 Configuration export and import	66
3.5 Post-implementation customization	68
3.5.1 Environmental variables	68
3.5.2 Directory ownership	69
3.5.3 ObjectServer creation and verification	69
3.5.4 Communication configuration	72
3.5.5 ObjectServer properties	76
Chapter 4. Configuration	85
4.1 Components configuration	86
4.1.1 Remote desktop installation	86
4.1.2 Gateway configuration	91
4.1.3 Process agent configuration	100
4.1.4 Startup script	104
4.2 Security configuration	106
4.2.1 Role creation	107
4.2.2 Group creation	112
4.2.3 User creation	115
4.3 ObjectServer customization	117
4.3.1 View creation	117
4.3.2 Filters definition	120

4.3.3 Menu creation	124
4.3.4 Database field definition	128
4.3.5 Tool definition	133
4.3.6 Trigger creation	140
4.3.7 Class creation	143
4.4 Probe configuration	144
4.4.1 Probe configuration sample	144
4.4.2 Probe property	146
4.4.3 Probe rule file	150
4.5 ObjectServer to ObjectServer communication	154
4.5.1 Bi-directional gateway	155
4.5.2 Uni-directional gateway	157
4.6 Accelerated Event Notification client	159
4.7 IBM Tivoli Health Monitoring agent for ObjectServer V7.2	171
Chapter 5. Operation	173
5.1 Device definition in simnet	174
5.2 Failover operation	176
5.3 Problem determination	178
5.3.1 Startup problems	178
5.3.2 Probe connection	180
5.3.3 Desktop startup	181
5.3.4 Slow response time	181
5.4 Performance tuning	184
Appendix A. Sample test	187
Sample questions	188
Answer key	191
Abbreviations and acronyms	193
Related publications	195
IBM Redbooks	195
Other publications	195
Online resources	195
How to get Redbooks	196
Help from IBM	196
Index	197

Archived

Figures

2-1 Overall components	38
2-2 No failover	45
2-3 Failover architecture	46
2-4 Desktop ObjectServer	47
2-5 Tiered ObjectServer implementation	48
2-6 SSL communication	51
2-7 Desktop communication	52
3-1 IBM Tivoli Netcool/OMNIBus event list login window on UNIX	71
3-2 IBM Tivoli Netcool/OMNIBus event list window on UNIX	72
3-3 The nco_xigen window	74
3-4 Interface definitions tool for IBM Tivoli Netcool/OMNIBus on Windows	75
3-5 IBM Tivoli Netcool/OMNIBus availability test on Windows	76
4-1 IBM Tivoli Netcool/OMNIBus installation	86
4-2 IBM Tivoli Netcool/OMNIBus installation License Agreement	87
4-3 IBM Tivoli Netcool/OMNIBus installation home location	87
4-4 IBM Tivoli Netcool/OMNIBus features selection	88
4-5 IBM Tivoli Netcool/OMNIBus installation	89
4-6 IBM Tivoli Netcool/OMNIBus installation complete	89
4-7 IBM Tivoli Netcool/OMNIBus Server Editor	90
4-8 IBM Tivoli Netcool/OMNIBus Windows Event List login window	90
4-9 IBM Tivoli Netcool/OMNIBus Windows Even list default window	91
4-10 AEL on both servers	98
4-11 AEL on NCOMS	99
4-12 AEL on NCOMS2	100
4-13 IBM Tivoli Netcool/OMNIBus Administrator Roles tab	107
4-14 IBM Tivoli Netcool/OMNIBus Administrator New Role tab	108
4-15 Permission tab	109
4-16 Permission Objects	110
4-17 Permissions tab	111
4-18 IBM Tivoli Netcool/OMNIBus Administrator Groups tab	112
4-19 IBM Tivoli Netcool/OMNIBus Administrator adding groups	113
4-20 IBM Tivoli Netcool/OMNIBus Administrator Restriction Filter tab	114
4-21 IBM Tivoli Netcool/OMNIBus Administrator windows Users tab	115
4-22 IBM Tivoli Netcool/OMNIBus Administrator Add User	116
4-23 Event List window	118
4-24 View Builder view	119
4-25 New view created	120
4-26 Filter builder button window	121

4-27	Filter Builder window	122
4-28	Restriction filter definition	123
4-29	Restriction Filter creation	124
4-30	Administrator console Menus tab	125
4-31	Menu definition	126
4-32	Administrator console Menus tab	127
4-33	Event list tools	128
4-34	Administrator console system console	129
4-35	New column window	130
4-36	Administrator console column added	131
4-37	Event list column	132
4-38	Administrator console Menu tab	133
4-39	Tool creation window	134
4-40	Tool creation window	135
4-41	Tool creation window	136
4-42	Tool creation window	137
4-43	Prompt.	138
4-44	Prompt definition	139
4-45	Triggers	140
4-46	Database trigger creation settings tab	141
4-47	Administrator console Visual window	143
4-48	Add Class window	144
4-49	IBM Tivoli Netcool/OMNIBus, Event List	146
4-50	Backup trigger enabled	156
4-51	Primary trigger group disabled on backup server	157
4-52	AEN icon	159
4-53	AEN client property: Application tab	160
4-54	Messages tab	161
4-55	AEN Channels tab	162
4-56	AEN login window	163
4-57	Create trigger action	164
4-58	Administration console Channels tab	165
4-59	Channel Details tab	166
4-60	Channel column details	167
4-61	Recipient created	168
4-62	Channel recipient window	169
4-63	Test channel	170
4-64	Test message	171
4-65	Health Monitoring agent	172
5-1	Simnet events	175
5-2	NCOMS_P events	176
5-3	Disconnect from ObjectServer	176
5-4	Failover	177

5-5 Backup ObjectServer event	177
5-6 Reconnect or failback message	178
5-7 Desktop connections.....	183

Archived

Archived

Tables

2-1 Roles.....	41
2-2 Groups	43
2-3 Users.....	44
2-4 Table name	53
2-5 The alerts.status table.....	56
3-1 ObjectServer properties	77
4-1 Common gateway properties	92
4-2 Bi-directional gateway properties	94
4-3 Probe property parameters	147

Archived

Examples

3-1 Home directory	63
3-2 License agreement	63
3-3 Installation feature selection	64
3-4 Installation completed	64
3-5 Probe installation	65
3-6 FixPack installation	66
3-7 nco_confpack -list	67
3-8 nco_confpack -export	67
3-9 nco_confpack -import	67
3-10 Environmental variables for IBM Tivoli Netcool/OMNIBus	68
3-11 Group creation on UNIX	69
3-12 IBM Tivoli Netcool/OMNIBus availability test on UNIX	70
3-13 Interface definitions file for IBM Tivoli Netcool/OMNIBus	73
3-14 Interface definition file for virtual ObjectServer	73
4-1 UNI_GATE.props	92
4-2 Additions to the interface definitions file	96
4-3 Sample mapping	97
4-4 nco_pa.conf, nco_process	101
4-5 nco_pa.conf, nco_routing	101
4-6 Sample nco_service	102
4-7 omni.dat gateway configuration	102
4-8 nco_pad	103
4-9 nco_pa_status	104
4-10 Startup script installation	105
4-11 The nco startup script variables	105
4-12 Startup symbolic links	105
4-13 nco start and nco_pa_status command	106
4-14 SQL statement to define a trigger	142
4-15 simnet.prop	145
4-16 simnet.rules	145
4-17 Verifying the probe is running	145
4-18 Switch example	151
4-19 Include sample	151
4-20 Discard and regmatch example	151
4-21 Lookup table	152
4-22 rules file table definition	152
4-23 Refreshing the probe	152
4-24 Installing Probe Rules Syntax Checker	153

4-25	Syntax check	153
4-26	The omni.dat file for two ObjectServers	155
4-27	omni.dat	158
4-28	Starting the desktop ObjectServer	158
4-29	Probe rule	163
4-30	importing schema	171
5-1	simnet.def	174
5-2	simnet.props	174
5-3	Checking \$NCHOME	178
5-4	Checking ObjectServer database	179
5-5	Checking the interface file	179
5-6	Checking port usage	179
5-7	Some probes error messages	180
5-8	Connection refused	180
5-9	Connection working	180
5-10	Setting DISPLAY variable	181
5-11	Checking number of events	181
5-12	Profile log	182
5-13	Finding the event rate	183
5-14	Checking desktop connections	184

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 illustrate 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.

Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

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

AIX®
DB2®
Foundations™
IBM®
Lotus®

Netcool®
PartnerWorld®
Rational®
Redbooks®
Redbooks (logo) ®

Tivoli Enterprise Console®
Tivoli®
ValueNet®
WebSphere®

The following terms are trademarks of other companies:

ValueNet, and the FileNet logo are registered trademarks of FileNet Corporation in the United States, other countries or both.

ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.

IT Infrastructure Library, IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

Oracle, JD Edwards, PeopleSoft, Siebel, and TopLink are registered trademarks of Oracle Corporation and/or its affiliates.

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

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

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

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

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

Preface

This IBM® Redbooks® publication is a study guide for the IBM Tivoli Netcool/OMNIbus V7.2 implementation certification test. It is aimed at the IT professional who wants to be an IBM Certified Professional for this product.

The IBM Tivoli Netcool/OMNIbus V7.2 certification test is offered through the IBM Professional Certification program. It is designed to validate the skills required of technical professionals who work in the implementation and deployment of IBM Tivoli Netcool/OMNIbus V7.2.

This book provides the necessary information for understanding the subject matter. It includes sample questions, which will help you evaluate your progress. It familiarizes the readers with the types of questions that may be encountered in the exam.

This guide does not replace practical experience, and it is not designed to be a stand-alone guide on this subject. Instead, this guide should be combined with educational activities and experiences and used as a useful preparation guide for the exam.

For your convenience, the chapters are based on the certification objectives of the IBM Tivoli Netcool/OMNIbus V7.2 implementation certification test. Those requirements are planning, prerequisites, installation, configuration, administration, and problem determination. Studying these chapters helps you prepare for the objectives of the exam.

The team who wrote this book

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

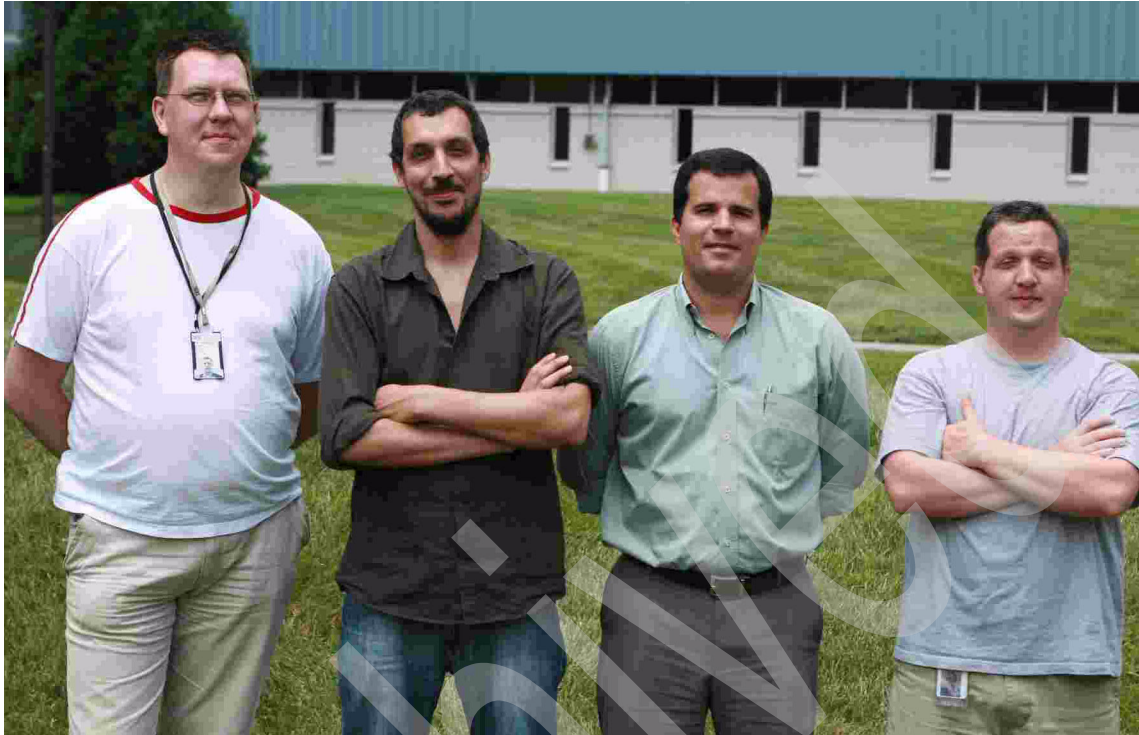


Figure 1 Mario, Thomas, Andre, and Phillip

Budi Darmawan is a Project Leader at the International Technical Support Organization, Austin Center. He writes extensively and teaches IBM classes worldwide on all areas of Tivoli® and systems management. Before joining the ITSO 10 years ago, Budi worked in Integrated Solution Services, IBM Indonesia as lead implementer and solution architect.

Thomas Boiocchi an IT Specialist based in Italy, working for Tivoli Services since 2007. He joined IBM after working for several years as a Netcool® Specialist for Eirteic Consulting travelling around the world. He has 10 years of IT experience and previously worked as a system and network administrator in Telkom and banks in Italy. His area of expertise include Omnibus, IBM Tivoli Business Services Manager, Network Manager, and Impact.

Andre Ricardo Cavalcanti de Araujo is a System Management Information Technology Specialist working with Tivoli Management Product in Brazil. He has 12 years of experience in servers and systems support. He hold a degree in Telecommunication Engineering and has MBA in Network Computer Management. His areas of expertise include UNIX/Linux® support, Cisco networking, networking security, and infrastructure and networking management.

Mario Schuerewegen is a Technical Presales specialist based in Belgium that specializes in Netcool products. He has 10 years of experience in the network and event management field. His areas of expertise include Cisco networking, SNMP, and network management.

Phillip Stanton is an L2 software support specialist based in the United States. He has 12 years of Information Technology experience and holds a Bachelors of Science degree in Business Administration with an emphasis on Information Systems. Most of his experience is in supporting and administrating mixed UNIX® and Microsoft® Windows® platforms for various Business to Business services and e-commerce Web sites. He joined IBM 2 years ago in L2 support for the Omnibus, Webtop, System Service monitors, and Internet Service Monitors software solutions. He currently holds the following certifications: Netcool Core V2, Netcool Core V3, Tivoli Level 2 Support Tools and Processes, and MCSE NT4/2000.

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

Tamikia Barrow and Margaret Ticknor
International Technical Support Organization, Raleigh Center

Wade Wallace
International Technical Support Organization, Austin Center

Jill Kanatzar
IBM Software Group, Worldwide Sales Channel Growth Executive

Become a published author

Join us for a two- to six-week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You will have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will 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 books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

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

ibm.com/redbooks

- ▶ Send your comments in an e-mail to:

redbooks@us.ibm.com

- ▶ Mail your comments to:

IBM Corporation, International Technical Support Organization
Dept. HYTD Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

Certification overview

This chapter provides an overview of the skills requirements needed to obtain an IBM Certified Deployment Specialist - IBM Tivoli Netcool/OMNIBus V7.2 certification. This chapter provides a comprehensive review of topics that are essential for obtaining the certification:

- ▶ 1.1, “IBM Professional Certification Program” on page 2
- ▶ 1.3, “Certification achieved” on page 26
- ▶ 1.2, “IBM Tivoli Netcool/OMNIBus V7.2 test objectives” on page 8
- ▶ 1.4, “Recommended study resources” on page 35

1.1 IBM Professional Certification Program

Having the right skills for the job is critical in the growing global marketplace. IBM Professional Certification is designed to validate skill and proficiency in the latest IBM solution and product technology. It can help provide that competitive edge.

The Professional Certification Program from IBM offers a business solution for skilled technical professionals seeking to demonstrate their expertise to the world.

The program is designed to validate your skills and demonstrate your proficiency in the latest IBM technology and solutions. In addition, professional certification may help you excel at your job by giving you and your employer the confidence that your skills have been tested. You may be able to deliver higher levels of service and technical expertise than non-certified employees and move on a faster career track.

The certification requirements are difficult, but are not overwhelming. Certification is a rigorous process that differentiates you from everyone else. The mission of IBM Professional Certification is to:

- ▶ Provide a reliable, valid, and fair method of assessing skills and knowledge
- ▶ Provide IBM with a method of building and validating the skills of individuals and organizations
- ▶ Develop a loyal community of highly skilled certified professionals who recommend, sell, service, support, and use IBM products and solutions

The Professional Certification Program from IBM has developed certification role names to guide you in your professional development. The certification role names include IBM Certified Specialist, IBM Certified Solutions/Systems Expert, and IBM Certified Advanced Technical Expert. These role names are for technical professionals who sell, service, and support IBM solutions. For technical professionals in application development, the certification roles include IBM Certified Developer Associate and IBM Certified Developer. An IBM Certified Instructor certifies the professional instructor.

The Professional Certification Program from IBM provides you with a structured program leading to an internationally recognized qualification. The program is designed for flexibility by allowing you to select your role, prepare for and take tests at your own pace, and, in some cases, select from a choice of elective tests best suited to your abilities and needs. Some roles also offer a shortcut by giving credit for a certification obtained in other industry certification programs.

You can be a network administrator, systems integrator, network integrator, solution architect, solution developer, value-added reseller, technical coordinator, sales representative, or educational trainer. Regardless of your role, you can start charting your course through the Professional Certification Program from IBM today.

You can learn more about the IBM Professional Certification Program by going to the following address:

<http://www.ibm.com/certify/index.shtml>

1.1.1 Benefits of certification

Certification is a tool to help objectively measure the performance of a professional on a given job at a defined skill level. Therefore, it is beneficial for individuals who want to validate their own skills and performance levels, their employees, or both. For optimum benefit, the certification tests must reflect the critical tasks required for a job, the skill levels of each task, and the frequency by which a task needs to be performed. IBM prides itself in designing comprehensive, documented processes that ensure that IBM certification tests remain relevant to the work environment of potential certification candidates.

In addition to assessing job skills and performance levels, professional certification can also provide such benefits as:

- ▶ For employees:
 - Promotes recognition as an IBM certified professional
 - Helps create advantages in interviews
 - Assists in salary increases, corporate advancement, or both
 - Increases self-esteem
 - Provides continuing professional benefits
- ▶ For employers:
 - Measures the effectiveness of training
 - Reduces course redundancy and unnecessary expenses
 - Provides objective benchmarks for validating skills
 - Makes long-range planning easier
 - Helps manage professional development
 - Aids as a hiring tool
 - Contributes to competitive advantage
 - Increases productivity
 - Increases morale and loyalty

Specific benefits can vary by country (region) and role. In general, after you become certified, you should receive the following benefits:

- ▶ Industry recognition

Certification may accelerate your career potential by validating your professional competency and increasing your ability to provide solid, capable technical support.

- ▶ Program credentials

As a certified professional, you receive, by way of e-mail, your certificate of completion and the certification mark associated with your role for use in advertisements and business literature. You can also request a hardcopy certificate, which includes a wallet-size certificate.

The Professional Certification Program from IBM acknowledges the individual as a technical professional. The certification mark is for the exclusive use of the certified individual.

- ▶ Ongoing technical vitality

IBM Certified professionals are included in mailings from the Professional Certification Program from IBM.

1.1.2 Tivoli Software Professional Certification

The IBM Tivoli Professional Certification program offers certification testing that sets the standard for qualified product consultants, administrators, architects, and Business Partners.

The program also offers an internationally recognized qualification for technical professionals seeking to apply their expertise in today's complex business environment. The program is designed for those who implement, buy, sell, service, and support IBM Tivoli solutions and want to deliver higher levels of service and technical expertise.

Benefits of being Tivoli certified

Tivoli certification provides the following benefits:

- ▶ For the individual:
 - IBM Certified certificate and use of logos on business cards
 - Recognition of your technical skills by your peers and management
 - Enhanced career opportunities
 - Focus for your professional development

- ▶ For the IBM Business Partner:
 - Confidence in the skills of your employees
 - Enhanced partnership benefits from the IBM Business Partner program
 - Billing your employees out at higher rates
 - Strengthens your proposals to customers
 - Demonstrates the depth of technical skills available to prospective customers
- ▶ For the customer:
 - Confidence in the services professionals handling your implementation
 - Ease of hiring competent employees to manage your Tivoli environment
 - Enhanced return on investment (ROI) through more thorough integration with Tivoli and third-party products
 - Ease of selecting a Tivoli Business Partner that meets your specific needs

Certification checklist

The certification process is as follows:

1. Select the certification that you want to pursue.
2. Determine which test or tests are required by reading the certification role description.
3. Prepare for the test, using the following provided resources:
 - Test objectives (1.2, “IBM Tivoli Netcool/OMNibus V7.2 test objectives” on page 8)
 - Recommended educational resources (1.4, “Recommended study resources” on page 35)
 - Sample/assessment test (Appendix A, “Sample test” on page 187)
 - Other reference materials
 - Opportunities for experience
4. Register to take a test by contacting one of our worldwide testing vendors:
 - Thomson Prometric
 - Pearson Virtual University Enterprises (VUE)
5. Take the test. Be sure to keep the Examination Score Report provided upon test completion as your record of taking the test.

6. Repeat steps three through five until all required tests are successfully completed for the desired certification role. If additional requirements are needed (such as another vendor certification or exam), follow the instructions on the certification description page to submit these requirements to IBM.
7. After you complete your certification requirements, you will be sent an e-mail asking you to accept the terms of the IBM Certification Agreement before receiving the certificate.
8. Upon acceptance of the terms of the IBM Certification Agreement, an e-mail will be sent containing the following electronic deliverables:
 - A Certification Certificate in PDF format, which can be printed in either color or black and white
 - A set of graphic files of the IBM Professional Certification mark associated with the certification achieved
 - Guidelines for the use of the IBM Professional Certification mark
9. To avoid unnecessary delay in receiving your certificate, ensure that we have your current e-mail on file by keeping your profile up to date. If you do not have an e-mail address on file, your certificate will be sent through postal mail.

After you receive a certificate by e-mail, you can also contact IBM at <mailto:certify@us.ibm.com> to request that a hardcopy certificate be sent by postal mail.

1.1.3 Growth through skills

Customers want to work with experts who understand their business and can help them achieve their objectives. IBM Business Partners who have expertise across the IBM software portfolio are well positioned to deliver high client value.

IBM Software will be announcing the next step in our Business Partner channel strategy focused on Growth Through Skills. In October 2009, IBM will roll out a new controlled distribution model to maximize value to our Business Partners and customers.

A subset of the IBM software portfolio will continue to be offered by way of the open distribution model or Software ValueNet®. The benefits of the growth through skills program are:

- ▶ Protects and maximizes your ROI in the technical, sales, and marketing skills you have developed.
- ▶ Places a premium on your skills and solutions that differentiate your ability to offer your customers guidance in a tough economy.

- ▶ Rewards the value you bring throughout the sales cycle by way of the lucrative IBM Software Value Incentive (SVI).
- ▶ Provides financial rewards for integrating IBM software with your business solutions by way of the Value Advantage Plus (VAP) incentive.
- ▶ Accelerates your growth with experienced software Value Added Distributors (VADs).
- ▶ Improves access to IBM resources, including industry-leading sales, technical, and marketing.

Authorization to resell IBM software products within controlled distribution is achieved at the product group level. There are 14 products groups across the five brands:

- ▶ WebSphere®
 - SOA Foundation
 - Connectivity
 - Business Process Management
 - Commerce
 - SOA Appliances
 - Enterprise Solutions (z)
- ▶ Tivoli
 - Storage Management
 - Security and Compliance Management
 - Automation
 - Enterprise Asset Management
- ▶ Information Management
 - Heritage CM
 - Data Management
- ▶ Lotus®
- ▶ Portal
- ▶ Rational®

The criteria for authorization to resell IBM Software products within controlled distribution include:

- ▶ Membership in the IBM PartnerWorld® program
- ▶ Approved participation in Software Value Incentive (SVI) or Value Advantage Plus (VAP)
 - For SVI, technical and sales skills in the product group(s) you want to sell
 - For VAP, an approved solution containing the product group(s) you want to sell
- ▶ An approved PartnerPlan
- ▶ Minimum revenue participation levels within SVI and VAP after the first year

IBM provides comprehensive enablement options to support the education, training, and certifications necessary to qualify for authorization to resell:

- ▶ Leverage the readiness assessment tools and work with your Distributor or IBM Business Partner Sales Representative to explore enablement opportunities and support.
- ▶ Visit the Subject Matter Expert (SME) Zone on the Virtual Innovation Center as a single point of entry to review software education and customized roadmaps.
- ▶ Learn about the You Pass, We Pay education reimbursement.
- ▶ Participate in readiness events throughout the year and revisit the Web for updates and the latest support materials.

1.2 IBM Tivoli Netcool/OMNibus V7.2 test objectives

The test has the following objectives:

- ▶ 1.2.1, “Planning” on page 9
- ▶ 1.2.2, “Installation” on page 9
- ▶ 1.2.3, “Configuration” on page 11
- ▶ 1.2.4, “Performance tuning and problem determination” on page 23
- ▶ 1.2.5, “Administration” on page 25
- ▶ 1.2.6, “Training” on page 25

For the most updated objectives of the IBM Tivoli Netcool/OMNibus V7.2 Implementation test (test #933), refer to the following address:

<http://www-03.ibm.com/certify/tests/obj933.shtml>

1.2.1 Planning

Given the customer requirements in the Statement of Work, review the solution details in order to confirm the architecture, with emphasis on performing the following steps:

1. Review the proposed architecture.
2. Determine the high level architecture.
3. Determine the unique requirements
4. Determine the hardware and OS.
5. Evaluate port availability (firewall and Access Control Lists)
6. Determine the contacts at the customer.
7. Gather users, roles, and physical location.
8. Evaluate the security requirements.
9. Determine the integrations to existing customer applications.

Given the design details, develop an architecture document, with emphasis on performing the following steps:

1. Verify all the software needed.
2. Determine the product integration probes and gateways.
3. Define the components, locations, and network connectivity.
4. Determine the naming conventions.
5. Determine the directory for the installation.
6. Review the customer compliance requirements (acceptance testing).
7. Determine the configuration of messages on the end nodes.

Given a design document, obtain the required components so that you are ready to install IBM Tivoli Netcool/OMNIBus, with emphasis on performing the following steps:

1. Determine the appropriate installer (console versus GUI).
2. Obtain network availability.
3. Obtain access to systems and servers.
4. Obtain the authorization for network.
5. Obtain the installation media.

1.2.2 Installation

Given that environment variables have been set and sourced on a supported UNIX/Linux server, install IBM Tivoli Netcool/OMNIBus V7.2 using the console so that the selected IBM Tivoli Netcool/OMNIBus V7.2 components are installed, with emphasis on performing the following steps:

1. Download the required IBM Tivoli Netcool/OMNIBus binaries from the IBM support site.

2. Place the binaries on the UNIX/Linux server in an appropriate temporary location.
3. Un-tar the package and run the installation script `./INSTALL -console`.
4. Accept the license agreement.
5. Select the required features:
 - Desktop
 - Gateways
 - Process Control
 - Servers
 - ConfPack
 - Administrator
 - AEN Client
 - Local Help System
 - Install selected features

Given that IBM Tivoli Netcool/OMNIBus is installed, download the probes patch from the IBM support site and install the patches using `$NCHOME/omnibus/install/nco_patch` so that the probe is installed, with emphasis on performing the following steps:

1. Download the probe patch from the IBM support site.
2. Untar the patch to a tmp directory.
3. Run `$NCHOME/omnibus/install/nco_patch install <path_to_patch_folder>`.
4. Type “yes” and press Enter to install the probe.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed on a UNIX host and a FixPack is downloaded from the IBM support site, install the FixPack, with emphasis on performing the following step:

Run the FixPack installation command `$NCHOME/install/ncisetup install <path_to_FixPack>`. You should receive a notice that the patch has been installed successfully.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured on UNIX/Linux, export the configuration of the ObjectServer and import it into another ObjectServer using `nco_confpack`, with emphasis on performing the following steps:

1. Create a list of exportable configuration items by running `$NCHOME/omnibus/bin/nco_confpack -list -server NCOMSI -file /tmp/NCOMSI_list`.
2. Edit the package list `/tmp/NCOMSI_list` so that it contains only the items to export.
3. Export the NCOMSI configuration by running `$NCHOME/omnibus/bin/nco_confpack -export -file /tmp/NCOMSI_list -package /tmp/NCOMSI_package`.
4. Import the NCOMSI configuration into NCOMS2 by running `$NCHOME/omnibus/bin/nco_confpack -import package /tmp/NCOMSI_package -server NCOMS2`.

1.2.3 Configuration

Given a supported UNIX/Linux operating system and IBM Tivoli Netcool/OMNIBus V7.2 and a user with proper permissions, configure and verify the environmental variables, with emphasis on performing the following steps:

1. Start the UNIX text editor.
2. Edit the system home profile `/etc/profile`.
3. Define Netcool Environment Variables by running `$NCHOME; $LD_LIBRARY_PATH (opt); $PATH (opt.); $LANG (if required)`.
4. Netcool users must source this file when they log in.

Given a UNIX/Linux OS, and ObjectServer was installed by the root user, the root user owns the `$NCHOME` directory. Configure the system so that a non-root user (`netcool`) is a member of the `netcool` group, with emphasis on performing the following steps:

1. Find an unused group number, for example, 550, by opening `/etc/group`.
2. Add a new group `netcool` by running `groupadd -g 550 netcool`.
3. Add the root user to the `netcool` group within `/etc/group`.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed on a IPv4 or IPv6 UNIX/Linux server, configure IBM Tivoli Netcool/OMNIBus communications, with emphasis on performing the following steps (The host name may or may not be resolvable to the IP address (IPv4 or IPv6)):

1. If the host name is resolvable to an IP address, edit `$NCHOME/etc/omni.dat` and replace “omnihost” with the local server’s host name. If the host name is not resolvable to an IP address, replace “omnihost” with the appropriate IPv4 or IPv6 address.
2. Run `$NCHOME/bin/nco_igen`.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured and environmental variables have been set on a supported Windows server, verify that IBM Tivoli Netcool/OMNIBus communication is available, with emphasis on performing the following steps:

1. Run the Netcool Server Editor by selecting **Run** → **All Programs** → **Netcool OMNIBUS** → **System Utilities** → **Server Editor**.
2. Highlight the appropriate ObjectServer and select **Test** in the Servers Available window.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed on UNIX/Linux, create a working object server instance using the available Netcool commands and files, so that an ObjectServer has been created and an instance configured, with emphasis on performing the following steps:

1. Open a command line and run `$NCHOME/omnibus/bin/nco_dbinit -server <name>`.
2. Edit the `$NCHOME/etc/omni.dat` file and set the ObjectServer name and port.
3. Run `$NCHOME/bin/nco_igen`.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed on UNIX/Linux, start the ObjectServer and run `nco_ping` to verify that the ObjectServer is running, with emphasis on performing the following steps:

1. Run `$NCHOME/omnibus/bin/nco_objserv -name <ObjectServer name>`.
2. Verify the ObjectServer is running by running `$NCHOME/omnibus/bin/nco_ping <ObjectServer name>`.

Given an ObjectServer is installed and running on a supported UNIX/Linux OS, start and verify a local event list so that local event lists can connect to the local ObjectServer, with emphasis on performing the following steps:

1. Launch the event list by running `$NCHOME/omnibus/bin/nco_event`.
2. Enter the user name and password.
3. Verify that the event list connects to the ObjectServer.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured and environmental variables have been set, install the remote desktop so that the remote desktop is on a supported Windows desktop and can connect to the ObjectServer, with emphasis on performing the following steps:

1. Locate and unzip the IBM Tivoli Netcool/OMNIBus binaries on the remote desktop machine.
2. Run the IBM Tivoli Netcool/OMNIBus install package (setup.exe).
3. Accept the license agreement.
4. Choose the installation directory.
5. Deselect every program feature except the desktop component.
6. Click the **Installation** button.
7. Reboot the desktop when the installation completes.
8. Create an entry for the ObjectServer in the Servers Editor on the remote desktop.
9. Launch the event list by selecting **Start** → **All Programs** → **Netcool Suite** → **Event List** and log into the ObjectServer.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed, edit the probe properties file, set the SERVER property to the name of the ObjectServer, and configure a probe to connect to an ObjectServer, so that the Probe is configured, with emphasis on performing the following steps:

1. Open the probe properties file.
2. Set the server property to the name of the ObjectServer.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured, the environmental variables have been set and sourced on a supported UNIX/Linux server, and the simnet probe package is installed, start and verify the probes so that the probes are running and tested, with emphasis on performing the following steps:

1. Modify the simnet.props file and verify the ObjectServer target in \$NCHOME/omnibus/probes/<arch>.
2. Modify the rules file (simnet.rules) to include the host system name in the manager field.
3. Start the probe by running **nco_p_simnet**.
4. Check the process list to verify that the probe is running.
5. Launch the event list and verify the probe events in the ObjectServer and verify that the events parsed correctly.
6. Check for errors in the log file.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed, configure an uni-directional gateway, with emphasis on performing the following steps:

1. Create a directory for gateway files by running **`$NCHOME/omnibus/gates/UNI_GATE`**.
2. Copy all the files from `$NCHOME/omnibus/gates/objserv_uni` to `$NCHOME/omnibus/gates/UNI_GATE`:
 - `objserv_uni.props`
 - `objserv_uni.map`
 - `objserv_uni.reader.tblrep.def`
 - `objserv_uni.startup.cmd`
3. Rename all the copied files to match the unique name of the uni-directional gateway, that is, `UNI_GATE.*`.
4. Edit the `UNI_GATE.props` file and modify the following properties:
 - Name
 - PropsFile
 - Gate.MapFile
 - Gate.StartupCmdFile
 - Gate.Reader.Server
 - Gate.Reader.TblReplicateDefFile
 - Gate.Writer.Server
5. Edit the `UNI_GATE.map` file and `UNI_GATE.reader.tblrep.def` file, and modify the entries to define which objectserver tables and fields are accessed by the gateway.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed, configure an Oracle® Remedy Gateway or ODBC gateway so that the uni-directional gateway has been configured.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed, both ObjectServers (that will be connected by way of the gateway) are running, start an ObjectServer gateway. Update an event in an ObjectServer (the source ObjectServer if a uni-directional gateway is used), confirm that the same event in the other ObjectServer has similarly changed, so that the gateway has been configured correctly and initialized, with emphasis on performing the following steps:

1. Start a ObjectServer gateway by running **`$NCHOME/omnibus/bin/nco_g_objserv_uni - name uni`** or **`$NCHOME/omnibus/bin/nco_g_objserv_bi - name BIGATE`**.

2. Verify that the gateway is working by deleting and or modifying an event on one ObjectServer and checking to see that the modification is made on the other ObjectServer.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed, add a server entry for `nco_pa` in `omni.dat`, run `nco_igen`, edit `nco_pa.conf`, set the host name and name of ObjectServer in `nco_pa.conf`, and configure process control, with emphasis on performing the following steps:

1. Open the `$NCHOME/omnibus/etc/nco_pa.conf` file.
2. Change the name of the ObjectServer from `NCOMS` (if different).
3. Set the `omnihost` value to the host name of the local machine under the `nco_routing` entry.
4. Set the user and password values under `nco_routing` if using secure mode; otherwise, delete the user and password values.
5. Add an entry to `$NCHOME/etc/omni.dat` for `nco_pa` and run `$NCHOME/bin/nco_igen`.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed on a UNIX/Linux system and process control is configured, run `nco_pad`, verify the PA status by running `nco_pa_status` to start process control, and verify that process control is available and running, with emphasis on performing the following steps:

1. Run `$NCHOME/omnibus/bin/nco_pad`.
2. Run `$NCHOME/omnibus/bin/nco_pa_status -server NCO_PA`.
3. Manually kill one of the running processes by running `kill pid`, where `pid` is the process ID of the process that was killed.
4. Run `$NCHOME/omnibus/bin/nco_pa_status -server NCO_PA` to verify that the killed process is running with a different process ID.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed on UNIX/Linux and root level access has been granted, configure, run, and verify the startup scripts, with emphasis on performing the following steps:

1. Execute the start-up script `$NCHOME/omnibus/install/startup/<arch>install`.
2. Verify the creation of symbolic links.
3. Input/verify the process control name.
4. Select secure mode or not.
5. Enter a value for the `netcool_license_file` variable. Save the file.
6. Start the `nco` script by running `nco start`.
7. Verify that the process control has started.
8. Run `nco stop`. Verify that `nco_pad` has stopped.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed on UNIX/Linux, copy all the gateway files from `$NCHOME/omnibus/gates/objserv_bi` to a new directory, configure the gateway properties file, configure a server entry for the bi-directional gateway, and create a bi-directional gateway between two ObjectServers, with emphasis on performing the following steps:

1. Create a directory for gateway files, for example, `$NCHOME/omnibus/gates/BI_GATE`.
2. Copy all files from `$NCHOME/omnibus/gates/objserv_bi` to `$NCHOME/omnibus/gates/BI_GATE`.
3. Edit `$NCHOME/omnibus/gates/BI_GATE/objserv_bi.props` and set the following properties:
 - Gate.MapFile
 - Gate.StartupCmdFile
 - Gate.ObjectServerA.Server
 - Gate.ObjectServerA.TblReplicateDefFile
 - Gate.ObjectServerB.Server
 - Gate.ObjectServerB.TblReplicateDefFile
4. Copy `$NCHOME/omnibus/gates/BI_GATE/objserv_bi.props` to `$NCHOME/omnibus/etc/BI_GATE.props`.
5. Edit `$NCHOME/etc/omni.dat` and add an entry for `BI_GATE`.
6. Run `$NCHOME/bin/nco_igen`.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured, add a user so that users are created in IBM Tivoli Netcool/OMNIBus, with emphasis on performing the following steps:

1. From the main IBM Tivoli Netcool/OMNIBus Administrator window, select the **User** tab and then select **User**.
2. Click the **Add User** icon in the toolbar.
3. Enter a Name and select an unused User ID.
4. Enter a Full Name.
5. Check the **Create Conversion** check box.
6. From the Groups tab, double-click **Administrator** to assign the administrator role. Click **OK**.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured and the environmental variables have been set, create a view based on the statement of work, with emphasis on performing the following steps:

1. Launch View Builder from the Event List.
2. Select **File** → **New**.
3. Enter the View Name.
4. From Available Field pane, select **Node**, **Severity**, and **Summary**.
5. From the Available Sort Fields pane, select **Severity**.
6. In the Sorted by pane, double-click the **Severity** icon to set the sort order to descending order.
7. Click **Apply**.
8. Click **Close**.
9. Select **File** → **Save** from Main Event List window.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured, use the Event List GUI (nco_event) to create a filter using the Desktop client, with emphasis on performing the following steps:

1. Click the **Filter Builder** button in Sub-Event List.
2. Click **File** → **New**.
3. Provide a Filter Name.
4. Select **Severity** from the Column drop-down menu.
5. Select **Greater than or Equal to** from the Operator drop-down menu.
6. Select **Minor (#)** from the Value drop-down menu.
7. Click **Apply**.
8. Click **File** → **Save** from the Main Event List window.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured, use the Administrator GUI (nco_config) to add a restriction filter, with emphasis on performing the following steps:

1. From the main IBM Tivoli Netcool/OMNIBus Administrator window, select the **Users** tab and then select **Restriction Filters**.
2. Select **Add Restriction Filter** from the toolbar.
3. Assign a Name to the filter.
4. Enter the filter criteria.
5. Click **OK**.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured, create a new menu, with emphasis on performing the following steps:

1. From the Menu tab, select **Add New Item** within the Configuration Manager GUI.
2. Select **Menu Item Type**, **Tool**, and **Title**, and then click **OK**.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured, use the Administrator GUI (nco_config) to create a new database field, with emphasis on performing the following steps:

1. Select the **System** tab from the Administrator GUI.
2. Select **Databases**.
3. Select **Databases Alerts** and **Table Status**.
4. Select the **Add Column** icon from the menu.
5. Enter Column Name and Data Type.
6. Click **OK**.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured, use the Administrator GUI (nco_config) to add a tool, with emphasis on performing the following steps:

1. In the Tools menu, click **Add Tool**.
2. Enter a Name.
3. Select **Enter Relevant Tool Instructions**.
4. Click **OK**.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured, use the Administrator GUI (nco_config) to create a trigger, with emphasis on performing the following steps:

1. Select **DB Trigger**.
2. Select **Trigger Group**.
3. Enter Trigger Name.
4. Set Trigger Priority.
5. Set Action on Delete.
6. Set the Apply To pane to Statement.
7. Enter the SQL code for the action:

```
Begin Write into dellogs1 values ('The following row was deleted at;  
getdate (), old.Node, old.Summary); End
```
8. Enable **Trigger**.
9. Click **OK**.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured and the ObjectServer is running, customize the Rules file so that a customized probe rules file is available, with emphasis on performing the following steps:

1. Open the probe rules file with text editor.
2. Create a temporary element to hold a value extracted from the @Summary field.
3. Assign a probe property to the @Summary field.
4. Add a “switch” statement to execute a rule based on node name.
5. Use an “include” statement to include a new rule file segment.
6. Make a change so that @AlertGroup is updated on deduplication.
7. Discard events with the word “test” in the @Summary field.
8. Create a lookup table to look up department name based on node name. Turn on probe details if @Summary=unknown.
9. Test the syntax of the new rules file and debug.
10. Update the probe rules file without stopping the probe.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed and running on UNIX/Linux, configure the primary ObjectServer (NCOMS_P), create a backup ObjectServer (NCOMS_B), set the BackupObjectServer property to TRUE, configure the bi-directional gateway, configure triggers and procedures, and create a high availability architecture, so that automated failover and failback architecture is configured, with emphasis on performing the following steps:

1. On the primary ObjectServer host, create the primary ObjectServer.
2. On the backup ObjectServer host, create the backup ObjectServer.
3. Edit \$NCHOME/omnibus/etc/NCOMS_B.props and set the Backup ObjectServer property to “True”.
4. Configure a bi-directional gateway between the primary and backup ObjectServers.
5. On the primary ObjectServer host, configure \$NCHOME/etc/omni.dat for the primary and backup ObjectServer entries.
6. On primary ObjectServer host, run **\$NCHOME/bin/nco_igen**. Copy the \$NCHOME/etc/interfaces.<arch> file to the backup ObjectServer host.
7. On the backup ObjectServer, enable “backup_startup, backup_counterpart_down, backup_counterpart_up” triggers for automation failover and failback.
8. On the backup ObjectServer, disable the “primary_only” trigger group for automation failover and failback.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured, add entries to `omni.dat` for the primary and desktop ObjectServers, run the command `$NCHOME/bin/nco-igen`, copy the `$NCHOME/etc/interfaces.<arch>` file from the primary ObjectServer host to the desktop ObjectServer host, configure the desktop ObjectServer, and configure a uni-directional gateway between the primary and desktop ObjectServers to set up the desktop architecture so that the desktop architecture is configured, with emphasis on performing the following steps:

1. On the primary ObjectServer host, edit `$NCHOME/etc/omni.dat` and add entries for the primary and desktop ObjectServers.
2. Run `$NCHOME/bin/nco_igen` and copy the `$NCHOME/etc/interfaces.<arch>` file from the primary ObjectServer host to the desktop ObjectServer host.
3. Create and configure the primary ObjectServer.
4. Create the desktop ObjectServer by running `$NCHOME/omnibus/bin/nco_dbinit -desktopserver -server DESKOS`.
5. Start the desktop ObjectServer.
6. Run `$OMNIHOME/bin/nco_sq1` to insert the following row in to the master national table or the DESKOS:
(0, MASTERS, 1)
7. Configure a uni-directional gateway between the primary ObjectServer and the desktop ObjectServer.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured on UNIX/Linux, export the configuration of the ObjectServer and import it into another ObjectServer by running `nco_confpack`, so that the ObjectServer configuration can be exported and imported, with emphasis on performing the following steps:

1. Create a list of exportable configuration items by running `$NCHOME/omnibus/bin/nco_confpack -list -server NCOMSI -file /tmp/NCOMSI_list`.
2. Edit the package list `/tmp/NCOMSI_list` so that it contains only the items that will be exported.
3. Export the NCOMSI configuration by running `$NCHOME/omnibus/bin/nco_confpack -export -file /tmp/NCOMSI_list -package /tmp/NCOMSI_package`.
4. Import the NCOMSI configuration in to NCOMS2 by running `$NCHOME/omnibus/bin/nco_confpack -import package /tmp/NCOMSI_package -server NCOMS2`.

Given IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured and the environmental variables have been set, add a group, with emphasis on performing the following steps:

1. From the main IBM Tivoli Netcool/OMNIBus Administrator window, select the **Users** tab and then select **Groups**.
2. Click the **Add Group** icon.
3. Assign a name to the group.
4. Assign an unused Group ID.
5. Assign a Role.
6. Assign a Restriction Filter.
7. Click **OK**.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured, create a class, with emphasis on performing the following steps:

1. Using the Configuration Manager, select **Add Class** from the Visual menu.
2. Assign an identifier.
3. Name the class.
4. Click **OK**.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured, create a role, with emphasis on performing the following steps:

1. Within the Configuration Manager, select the **Users** tab.
2. Select **Roles** and the **Add New Role** menu item.
3. Add a Role name.
4. Assign Role Permissions.
5. Click **Save**.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed and running, configure the IBM Tivoli Netcool/OMNIBus probes rules to flag some events for accelerated event notification, create channels, and configure triggers to send event details using these channels so that accelerated event notification is configured for some events, with emphasis on performing the following steps:

1. Modify the probe rules file using the text editor and flag the events that have a summary, such as "Port failure".
2. Configure channels to broadcast event data for the flagged events.
3. Using IDUC EVTFT and IDUC SNDMSG, create and configure post-insert or post-update or post-reinsert triggers to send event notifications to the Accelerated Event Notification (AEN) client.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured on a UNIX/Linux system, configure the Accelerated Event Notification (AEN) client, with emphasis on the following steps:

1. Start the AEN client by running `$NCHOME/omnibus/bin/nco_aen`.
2. If you are starting the AEN for the first time, right-click the AEN icon and select **Properties**.
3. On the Application tab, make the following selections:
 - a. Notification settings:
 - b. Server (from drop-down list). If it is unpopulated, or the required server is not available, add details to the interfaces file.
 - c. View in either Netcool Event List or Netcool Webtop.
 - d. Launch in Control Settings.
 - e. For either Netcool Event List or Netcool Webtop, specify the required Server and View Browser Settings.
4. Specify the browser to use.
5. Configure the Message and Channels tabs, as required.
6. Select **OK**.
7. Given that the AEN client has previously been configured, log on to AEN client by right-clicking the AEN icon and select **Sign In**.
8. Input a valid ObjectServer user's details and select **Log In**.
9. Notice that the icon changes from x to y.

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed on a UNIX/Linux system, configure the ObjectServer system so that Accelerated Event Notification messages can be sent, with emphasis on the following steps:

1. Determine whether AEN events are to be generated by a probe or by a trigger.
2. If AEN events are to be generated by a probe:
 - a. Create a dedicated ObjectServer Flag field.
 - b. Modify the probe rules file to set the Flag field for AEN events.
 - c. Create a database trigger to respond to either the Flag field or the appropriate database action/database condition using the SQL commands IDUC EVTFT or IDUC SNDMSG to generate the AEN.

3. Configure a Channel to broadcast AEN event data by performing the following steps:
 - a. From with IBM Tivoli Netcool/OMNIBus Administrator, select **Channels** from the System menu.
 - b. Right-click in the GUI and select **Add a new Channel**.
 - c. Within the Channel Details pane, add a name for the Channel.
 - d. On the Channels tab, select the **Channel Columns Detail** button in order to define the ObjectServer Table and Fields to use for notification, and then select **OK**.
 - e. On the Recipients tab, select the **Add new Channel Recipient** button to be able to define recipients for the notification, and then select **OK**.
 - f. When complete, click the **OK** button on the Channel Details pane.
 - g. Test the Channel notification by right-clicking the Channel and selecting **Send Message**.

Given that ObjectServer V7.2 is running on a UNIX/Linux platform and the Tivoli Health Monitoring Agent for ObjectServer V7.2 is installed, configure the agent so that the agent will monitor the ObjectServer, with emphasis on the following steps:

1. Run `cd <tivoli installdir>/<arch>/no/bin`.
2. Import the schema and automations to the ObjectServer by running `$NCHOME/bin/nco_sql -user root -S<server_name> < itm_os.sql`.

1.2.4 Performance tuning and problem determination

Given that IBM Tivoli Netcool/OMNIBus V7.2 has been installed and configured, add devices to the `simnet.def` file and run the `simnet` probe to generate test events using the `simnet` probe, with emphasis on performing the following steps:

1. Edit the `$NCHOME/omnibus/probes/ARCH/simnet.def` file and add the line `device1 0 50 & device2 3 100`.
2. Set the probe definition property in `$NCHOME/omnibus/probes/ARCH/simnet.props` to `$NCHOME/omnibus/probes/ARCH/simnet.def`.
3. Set the server property in `$NCHOME/omnibus/probes/ARCH/simnet.props` to the name of the ObjectServer.
4. Start the `simnet` probe by running `$NCHOME/omnibus/probes/nco_p_simnet`.

Given a failover architecture, verify that the failover architecture is working, with emphasis on performing the following steps:

1. Shut down the Master ObjectServer.
2. Connect the desktop client to the backup ObjectServer by running **`$NCHOME/omnibus/bin/nco_event`** and check that the events are coming into the backup ObjectServer.
3. Restart the Master ObjectServer.
4. Connect the desktop client to the Master ObjectServer by running **`$NCHOME/omnibus/bin/nco_event`** and check that the client indicates that it is connected to the Master ObjectServer.

Given that ObjectServer V7.2 is installed on a UNIX/Linux OS but is unable to start, determine why the ObjectServer does not start, with emphasis on performing the following steps:

1. Check the environment variable `$NCHOME`.
2. Check to make sure that the ObjectServer has been created by running **`$NCHOME/omnibus/bin/nco_dbinit`**.
3. Check to make sure that the entry for the ObjectServer exists in `$NCHOME/etc/omni.dat` and in the interfaces file.
4. Ensure that the port for the ObjectServer is not already in use.
5. Ensure that the proper user is starting the object server process.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed and the license server configured, determine why probes do not connect, with emphasis on performing the following steps:

1. Check the probes log file.
2. Verify that the probe's designated ObjectServer is running.
3. Check whether the interfaces file on the probe server is correctly defined.
4. Verify that the probe server has network connectivity to the ObjectServer host.
5. Check whether any firewall settings are affecting communications.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed and the license server is configured, determine why the desktop does not start, with emphasis on performing the following steps:

1. Ensure that the server settings are set correctly.
2. Ensure that the ObjectServer to which the desktop is connected is running.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed and configured, users are complaining of slow response time. Determine the reason for slow event list response, with emphasis on performing the following steps:

1. Check the number of events.
2. Check the OS profiles.
3. Check the event rates.
4. Check the number of desktops.
5. Check the processes on the OS server.
6. Check the resource usage of the OS server, memory, CPU, and disk.
7. Check the network response times.
8. Check the log files.
9. Check the automations.
10. Check the number of journals and details.
11. Determine what other components are connected.

Given that IBM Tivoli Netcool/OMNIBus V7.2 is installed and configured, optimize system performance so that the ObjectServer performance is optimized, with emphasis on performing the following steps:

1. Check the frequency of triggers.
2. Check the execution scope of triggers (once only for each row).
3. Check the number of details by running `details($*)`.
4. Check the SQL used in the trigger for performance.
5. Check whether “update via” is being used.
6. Check the desktop filters.

1.2.5 Administration

Given a running and verified system, complete the post implementation process, including system backup, documentation, acceptance testing, so that customer acceptance can be obtained, with emphasis on performing the following steps:

1. Back up the installed system.
2. Create the system environment documentation.
3. Verify compliance with customer requirements.
4. Transfer knowledge to the staff.
5. Demonstrate the system to the client.
6. Complete acceptance testing.
7. Get customer sign-off.

1.2.6 Training

Given new users, provide training, so that users are educated on the new system, with emphasis on training users and administrators.

1.3 Certification achieved

The test IBM Tivoli Netcool/OMNIBus V7.2 implementation (#933) is a prerequisite for achieving the following certifications:

- ▶ 1.3.1, “Tivoli Netcool Core V3.0” on page 26
- ▶ 1.3.2, “Tivoli Netcool Impact V4.0” on page 29
- ▶ 1.3.3, “Tivoli Fault Management Solutions 2008” on page 31
- ▶ 1.3.4, “Tivoli Performance Management Solutions 2008” on page 32
- ▶ 1.3.5, “Tivoli Business Application Management Solutions 2008” on page 32
- ▶ 1.3.6, “IBM Service Management Network and Service Assurance 2009” on page 33
- ▶ 1.3.7, “IBM Service Management Data Center Management and Transformation 2009” on page 34

1.3.1 Tivoli Netcool Core V3.0

Tivoli Netcool Core V3.0 is an IBM Certified Deployment Professional certification.

Target audience

An IBM Certified Deployment Professional - Tivoli Netcool Core V3.0 is a technical professional responsible for the planning, installing, configuring performance tuning, problem determination, and documenting of solutions for IBM Tivoli Netcool/OMNIBus V7.2 and IBM Tivoli Netcool/Webtop V2.0. This individual will be expected to perform these tasks with limited assistance from peers, product documentation, and support resources.

Recommended prerequisite skills

The IBM Tivoli Netcool/OMNIBus V7.2 key areas of competency are:

- ▶ Describe the IBM Tivoli Netcool/OMNIBus V7.2 architecture and components.
- ▶ Plan and design an IBM Tivoli Netcool/OMNIBus V7.2 solution based on the customer’s requirements/environment.
- ▶ Install and configure the prerequisites to IBM Tivoli Netcool/OMNIBus V7.2.
- ▶ Install and configure IBM Tivoli Netcool/OMNIBus V7.2 infrastructure components (probes, gateways, desktops, process control, accelerated event notification (AEN), IPV6 configuration, and automated failover and failback).
- ▶ Use various interfaces to configure and administer the IBM Tivoli Netcool/OMNIBus V7.2 environment.

- ▶ Perform performance tuning and problem determination for IBM Tivoli Netcool/OMNIBus V7.2.

The IBM Tivoli Netcool/OMNIBus V7.2 required prerequisite(s) are:

- ▶ Strong working knowledge of the IBM Tivoli Netcool/OMNIBus V7.2 infrastructure components (ObjectServer, probes, gateways, desktop and process control, accelerated event notification (AEN), IPV6 configuration, and dual server desktop)
- ▶ Strong working knowledge of IBM Tivoli Netcool/OMNIBus V7.2 administration (triggers, procedures, tools, menus, automated failover and failback, AEN channels, and so on)
- ▶ Strong working knowledge of network management principles
- ▶ Working knowledge of operating system (UNIX and Windows), networking, and firewall concepts
- ▶ Knowledge of IBM Tivoli Netcool licensing
- ▶ Working knowledge of security (SSL and system user accounts)
- ▶ Working knowledge of SQL (including procedures)
- ▶ Working knowledge of scripting languages (shell scripting, Rules files, and regular expressions)
- ▶ Working knowledge of protocols, including TCP/IP and SNMP
- ▶ Working knowledge of operating system utilities (ftp, telnet, sftp, ssh, and text editor)

The IBM Tivoli Netcool/OMNIBus V7.2 recommended prerequisites are:

- ▶ Basic knowledge of help desk and database systems
- ▶ Knowledge of network management systems

The IBM Tivoli Netcool/Webtop V2.0 key areas of competency are:

- ▶ Describe the IBM Tivoli Netcool/Webtop V2.0 architecture and components.
- ▶ Plan and design an IBM Tivoli Netcool/Webtop V2.0 solution based on the customer's requirements/environment.
- ▶ Install and configure the prerequisites to IBM Tivoli Netcool/Webtop V2.0.
- ▶ Install and configure the IBM Tivoli Netcool/Webtop V2.0 infrastructure components (server, Webtop Administration API, event lists, maplets, and charts).
- ▶ Use various interfaces to configure and administer the IBM Tivoli Netcool/Webtop V2.0 environment.

- ▶ Perform performance tuning and problem determination for IBM Tivoli Netcool/Webtop V2.0.

The IBM Tivoli Netcool/Webtop V2.0 required prerequisites are:

- ▶ Strong working knowledge of IBM Tivoli Netcool/Webtop V2.0 infrastructure components
- ▶ Strong working knowledge of IBM Tivoli Netcool/Webtop V2.0 administration (tools, menus, maplets, entities, resources, users, filters, views, SMARTPage commands, and so on)
- ▶ Strong working knowledge of HTML, CGI, XML, and SQL
- ▶ Strong working knowledge of IBM Tivoli Netcool/OMNIBus V7.x, IBM Tivoli Netcool/Security Manager V1.3, IBM Tivoli Netcool/GUI Foundations™ V1.x, and associated architectures
- ▶ Working knowledge of IBM Tivoli/Netcool licensing
- ▶ Working knowledge of operating systems (UNIX and Windows), networking, and firewall concepts
- ▶ Working knowledge of security (SSL and system user accounts)
- ▶ Working knowledge of scripting languages (shell scripting and regular expressions)
- ▶ Working knowledge of Web browsers and Java™ applications

The IBM Tivoli Netcool/Webtop V2.0 recommended prerequisites are:

- ▶ Basic knowledge of network management systems
- ▶ Knowledge of Web application development techniques (JavaScript™ and HTML style sheets).
- ▶ Working knowledge of protocols, including TCP/IP.
- ▶ Working knowledge of operating system utilities (ftp, telnet, sftp, ssh, text editor, and so on)

Requirements

This certification requires two tests:

- ▶ Test 933 - IBM Tivoli Netcool/OMNIBus V7.2 Implementation
- ▶ Test 922 - IBM Tivoli Netcool/Webtop V2.0

1.3.2 Tivoli Netcool Impact V4.0

Tivoli Netcool Impact V4.0 is an IBM Certified Deployment Professional certification.

Target audience

An IBM Certified Deployment Professional - Tivoli Netcool Impact V4.0 is an individual who has demonstrated the ability to implement and support an IBM Tivoli Netcool Impact solution. It is expected that this person is able to perform the following tasks independently a majority of the time, and in some situations, take leadership and provide mentoring to peers. It is expected that this person will be able to perform these tasks with limited assistance from peers, product documentation, and vendor support services.

Recommended prerequisite skills

The key areas of competency are:

- ▶ Describe the IBM Tivoli Netcool Impact V4.0 architecture and components.
- ▶ Plan and design an IBM Tivoli Netcool Impact V4.0 solution based on the customer's requirements/environment.
- ▶ Install and configure prerequisites to IBM Tivoli Netcool Impact V4.0.
- ▶ Install and configure IBM Tivoli Netcool Impact V4.0 infrastructure components.
- ▶ Use available interfaces to configure and administer the IBM Tivoli Netcool Impact V4.0 environment.
- ▶ Perform performance tuning and problem determination for IBM Tivoli Netcool Impact V4.0.
- ▶ Develop and deploy IBM Tivoli Netcool Impact policies and services.

The required prerequisites are:

- ▶ Experience administering IBM Tivoli Netcool Impact V4.0 at skill level 4
- ▶ Knowledge of IBM Tivoli Netcool Impact V4.0 at skill level 4
- ▶ Knowledge of scripting languages (shell scripting, Rules files, regular expressions, Perl) at skill level 3
- ▶ Knowledge of operating systems (UNIX and Windows) at skill level 3
- ▶ Knowledge of networks and network management at skill level 3
- ▶ Knowledge of IBM Tivoli Netcool/OMNIBUS at skill level 3
- ▶ Knowledge of database structures at skill level 3

- ▶ Knowledge of operating system utilities (ftp, telnet, sftp, ssh, and text editors) at skill level 2
- ▶ Knowledge of SQL, POST-GRES, and other ANSI compliant sources at skill level 2
- ▶ Knowledge of HTML at skill level 2
- ▶ Knowledge of WebSphere Application Server Community Edition at skill level 1
- ▶ Knowledge of Web services (XML, SOAP, and WSDL), if applicable, at skill level 1
- ▶ Knowledge of Netcool Security Manager at skill level 1
- ▶ Knowledge of CVS at skill level 1

The skill descriptions are:

- | | |
|----------|---|
| 1 | Basic Skill/Knowledge: Familiarity with basic functionality and concepts. May need to rely on assistance from documentation or other resources. |
| 2 | Working Skill/Knowledge: Working knowledge of functionality and concepts. Can use products or explain concepts with little or no assistance. |
| 3 | Advanced Skill/Knowledge: Substantial experience with functionality or concepts. Can teach others how to use functionality or explain concepts. |
| 4 | Expert Skill/Knowledge: Extensive and comprehensive experience with functionality or concepts. Can create or customize code, architecture, or processes. |

This certification requires the IBM Tivoli Netcool/OMNIBus V7.x Implementation or Tivoli Enterprise Console® 3.9 Implementation certification as well as passing the IBM Tivoli Netcool Impact V4.0 Implementation exam.

Requirements

This certification requires two tests:

- ▶ Any one of the following tests:
 - Test 594 - IBM Tivoli Enterprise Console V3.9 Implementation
 - Test 901 - IBM Tivoli Netcool/OMNIBus V7.1 Implementation
 - Test 933 - IBM Tivoli Netcool/OMNIBus V7.2 Implementation
- ▶ Test 938 - IBM Tivoli Netcool Impact V4.0 Implementation

1.3.3 Tivoli Fault Management Solutions 2008

Tivoli Fault Management Solutions 2008 is an IBM Certified Advanced Deployment Professional certification.

Target audience

An IBM Certified Advanced Deployment Professional - Tivoli Fault Management Solutions 2008 is an individual who has demonstrated a higher level of implementation knowledge and skill both in breadth and in depth in the IBM Tivoli Fault Management solutions area.

Requirements

This certification requires four tests:

- ▶ Any one of the following tests:
 - Test 901 - IBM Tivoli Netcool/OMNIBus V7.1 Implementation
 - Test 933 - IBM Tivoli Netcool/OMNIBus V7.2 Implementation
- ▶ Test 922 - IBM Tivoli Netcool/Webtop V2.0
- ▶ Any two of the following tests:
 - Test 890 - IBM Tivoli Monitoring V6.1 Implementation
 - Test 897 - IBM Tivoli Network Manager IP Edition V3.7 Implementation
 - Test 905 - IBM Tivoli Composite Application Manager for WebSphere V6.1
 - Test 920 - IBM Tivoli Composite Application Manager for Response Time V6.2 Implementation
 - Test ITIL® - Information Technology Infrastructure Library - Foundations
 - Test 436 - IBM Tivoli Business Service Manager V4.1.1 Implementation
 - Test 938 - IBM Tivoli Netcool Impact V4.0 Implementation
 - Test 908 - IBM Tivoli Monitoring V6.2 Implementation

1.3.4 Tivoli Performance Management Solutions 2008

Tivoli Performance Management Solutions 2008 is an IBM Certified Advanced Deployment Professional certification.

Target audience

An IBM Certified Advanced Deployment Professional - Tivoli Performance Management Solutions 2008 is an individual who has demonstrated a higher level of implementation knowledge and skill both in breadth and in depth in the IBM Tivoli Performance Management solutions area.

Requirements

This certification requires four tests:

- ▶ Any one of the following tests:
 - Test 901 - IBM Tivoli Netcool/OMNIBus V7.1 Implementation
 - Test 933 - IBM Tivoli Netcool/OMNIBus V7.2 Implementation
- ▶ Test 922 - IBM Tivoli Netcool/Webtop V2.0
- ▶ Any two of the following tests:
 - Test ITIL - Information Technology Infrastructure Library - Foundations
 - Test 430 - IBM Tivoli Netcool Service Quality Manager V4.1.1 Implementation
 - Test 434 - IBM Tivoli Netcool Performance Manager for Wireless V9.1.2 Implementation
 - Test 931 - IBM Tivoli Netcool/Proviso V4.4.1 Implementation

1.3.5 Tivoli Business Application Management Solutions 2008

Tivoli Business Application Management Solutions 2008 is an IBM Certified Advanced Deployment Professional certification.

Target audience

An IBM Certified Advanced Deployment Professional - Tivoli Business Application Management Solutions 2008 is an individual who has demonstrated a higher level of implementation knowledge and skill both in breadth and in depth in the IBM Tivoli Business Application Management solutions area.

Requirements

This certification requires three tests:

- ▶ Any one of the following tests:
 - Test 594 - IBM Tivoli Enterprise Console V3.9 Implementation
 - Test 933 - IBM Tivoli Netcool/OMNibus V7.2 Implementation
- ▶ Test 908 - IBM Tivoli Monitoring V6.2 Implementation
- ▶ Any one of the following tests:
 - Test 253 - IBM WebSphere Application Server Network Deployment V6.1, Core Administration
 - Test 731 - DB2® 9 DBA for Linux UNIX and Windows
 - Test 905 - IBM Tivoli Composite Application Manager for WebSphere V6.1
 - Test ITIL - Information Technology Infrastructure Library -- Foundations

1.3.6 IBM Service Management Network and Service Assurance 2009

IBM Service Management Network and Service Assurance 2009 is an IBM Certified Advanced Deployment Professional certification.

Target audience

An IBM Certified Advanced Deployment Professional - IBM Service Management Network and Service Assurance 2009 is an individual who has demonstrated a higher level of implementation knowledge and skill both in breadth and in depth in the IBM Tivoli Network and Service Assurance solutions area.

Requirements

This certification requires four tests:

- ▶ Test 000-922 - IBM Tivoli Netcool/Webtop V2.0
- ▶ Test 000-933 - IBM Tivoli Netcool/OMNibus V7.2 Implementation
- ▶ Test 000-938 - IBM Tivoli Netcool Impact V4.0 Implementation
- ▶ Any one of the following tests:
 - Test ITIL - Information Technology Infrastructure Library -- Foundations
 - Test 000-430 - IBM Tivoli Netcool Service Quality Manager V4.1.1 Implementation
 - Test 000-434 - IBM Tivoli Netcool Performance Manager for Wireless V9.1.2 Implementation
 - Test 000-931 - IBM Tivoli Netcool/Proviso V4.4.1 Implementation

1.3.7 IBM Service Management Data Center Management and Transformation 2009

IBM Service Management Data Center Management and Transformation 2009 is an IBM Certified Advanced Deployment Professional certification.

Target audience

An IBM Certified Advanced Deployment Professional - IBM Service Management Data Center Management and Transformation 2009 is an individual who has demonstrated a higher level of implementation knowledge and skill both in breadth and in depth in the IBM Tivoli Data Center Management and Transformation solutions area.

Requirements

This certification requires three or four tests:

- ▶ Test 000-011 IBM Tivoli Application Dependency and Discovery Manager V7.1 Implementation
- ▶ Test 000-908 IBM Tivoli Monitoring V6.2 Implementation
- ▶ Any one (or two) of the following tests:
 - Test ITIL - Information Technology Infrastructure Library - Foundations
 - Test 000-012 IBM Tivoli Usage and Accounting Manager V7.1 Implementation
 - Test 000-253 IBM WebSphere Application Server Network Deployment V6.1 Core Administration
 - Test 000-435 IBM Tivoli Workload Scheduler V8.4 Implementation
 - Test 000-436 IBM Tivoli Business Service Manager V4.1.1 Implementation
 - Test 000-731 DB2 9 DBA for Linux UNIX and Windows
 - Test 000-905 IBM Tivoli Composite Application Manager for WebSphere V6.1
 - Test 000-920 IBM Tivoli Composite Application Manager for Response Time V6.2 Implementation
 - Test 000-922 IBM Tivoli Netcool/Webtop V2.0 and Test 000-933 IBM Tivoli Netcool/OMNIBus V7.2 Implementation

1.4 Recommended study resources

Courses and publications are offered to help you prepare for the certification tests. The courses are recommended, but not required, before taking a certification test. If you wish to purchase Web-based training courses or are unable to locate a Web-based course or classroom course at the time and location you desire, please feel free to contact one of our delivery management teams at:

- ▶ Americas: tivamedu@us.ibm.com
- ▶ EMEA: tived@uk.ibm.com
- ▶ AP: tivtrainingap@au1.ibm.com

Note that course offerings are continuously being added and updated. If you do not see a course listed in your geography, please contact the delivery management team.

1.4.1 Classroom courses

- ▶ Course title: IBM Tivoli Netcool/OMNIBus V7.1

Course duration: 1 day

Abstract: The IBM Tivoli Netcool/OMNIBus User course is designed to provide a basic understanding of the IBM Tivoli Netcool/OMNIBus Version 7.1 product and provide practical experience in how it may be used to manage events in a network computing environment. This User course is a prerequisite for the IBM Tivoli Netcool/OMNIBus Administration and Configuration course. The User course focuses on the everyday use of the IBM Tivoli Netcool/OMNIBus product from the user perspective. Each major feature will be highlighted and students will have their own system available to practice the tasks described and outlined in the course.

- ▶ Course title: IBM Tivoli Netcool/OMNIBus 7.2 Administration and Configuration

Course duration: 5 days

Abstract: A five day comprehensive training for operators and administrators starting with a basic understanding of the IBM Tivoli Netcool/OMNIBus Version 7.2 product, how it may be used to manage events in a network computing environment, followed by installation, administration, and configuration of the IBM Tivoli Netcool/OMNIBus V7.2 solution. Subjects explored include operator usage training, including basic fault management techniques using native V7.2 operator interface(s), installation and licensing procedures, configuring security and component communications, as well as

architecture and implementation. Examples of ObjectServer SQL syntax and automations are examined. The main focus is on the deployment, configuration, and everyday maintenance of IBM Tivoli Netcool/OMNIBus V7.2 from the perspective of the administrator. At the outset of the training, students will learn basic operational characteristics and engage in an installation of IBM Tivoli Netcool/OMNIBus V7.2 on a classroom server, followed by covering the standard configuration objects through classroom exercises.

1.4.2 Online course

- ▶ Course title: IBM Tivoli Netcool/OMNIBus 7.2 User

Course duration: 4 hours

Abstract: The IBM Tivoli Netcool/OMNIBus User course provides a basic understanding of IBM Tivoli Netcool/OMNIBus Version 7.2 product and practical experience in using it to manage events in a network computing environment. This course focuses on the everyday use of the IBM Tivoli Netcool/OMNIBus product from the user's perspective. Each major feature will be highlighted.

Planning

This chapter discusses some planning issues with IBM Tivoli Netcool/OMNIbus. The topics discussed are:

- ▶ 2.1, “IBM Tivoli Netcool/OMNIbus components” on page 38
- ▶ 2.2, “IBM Tivoli Netcool/OMNIbus configuration” on page 45
- ▶ 2.3, “Configuration planning issues” on page 49
- ▶ 2.4, “ObjectServer tables” on page 52

2.1 IBM Tivoli Netcool/OMNibus components

This section discusses the following topics:

- ▶ 2.1.1, “Architectural components” on page 38
- ▶ 2.1.2, “Desktop and administrative components” on page 40
- ▶ 2.1.3, “Groups, roles, and users” on page 41
- ▶ 2.1.4, “Insert Delete Update Cycle (IDUC) components” on page 44

2.1.1 Architectural components

The components of IBM Tivoli Netcool/OMNibus are shown in Figure 2-1.

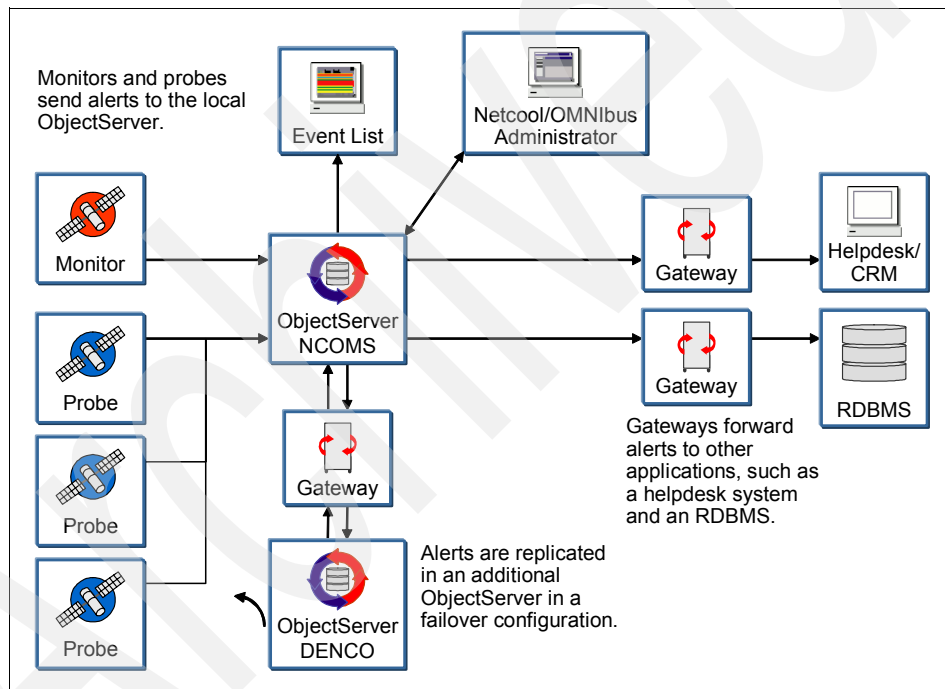


Figure 2-1 Overall components

Where:

- ▶ ObjectServer

The ObjectServer is the database server at the core of IBM Tivoli Netcool/OMNIBus. Event information is forwarded to the ObjectServer from external programs, such as probes, monitors, and gateways. The ObjectServer stores and manages this information in database tables, and displays the information in the event list.

- ▶ Probes

Probes connect to an event source, detect and acquire event data, and forward the data to the ObjectServer as alerts. Probes use the logic specified in a rules file to manipulate the event elements before converting them into the fields of an alert in the ObjectServer alerts.status table. Each probe is uniquely designed to acquire event data from a specific source. Probes can acquire data from any stable data source, including devices, databases, and log files.

- ▶ Monitors

Monitors are programs that acquire a resource's status and forwards the status to ObjectServer as alerts.

- ▶ Gateway

IBM Tivoli Netcool/OMNIBus gateways enable you to exchange alerts between ObjectServers and other applications, such as databases or help desk or Customer Relationship Management (CRM) systems. You can use gateways to replicate alerts or to maintain a backup ObjectServer. Application gateways enable you to integrate different business functions. For example, you can configure a gateway to send alert information to a help desk system. You can also use a gateway to archive alerts to a database. A gateway that allows forwarding of alerts to external network management systems is the SNMP gateway.

By default, IBM Tivoli Netcool/OMNIBus ObjectServer uses port 4100 to communicate with other components. Gateway and process controls need a port to be configured to work correctly. This can be done by adding information inside the interface file, which contains a list of IBM Tivoli Netcool/OMNIBus components and the ports they are using.

2.1.2 Desktop and administrative components

Apart from the components discussed in 2.1.1, “Architectural components” on page 38, there are some components that allow administrative and control functions to be performed. These components are also shown in Figure 2-1 on page 38. These components are:

- ▶ Administration console

IBM Tivoli Netcool/OMNIbus Administrator is a graphical tool that you can use to configure and manage ObjectServers.

- Menu and tools

An administrator can modify a specific menu layout and tools set to any specific users or group in order to provide a customized working environment to event list operators.

Tools add the capability to control alert management functions within the event list. New tools can be created and belong to two categories:

- SQL tools: Tool where update, insert, and delete are executed on the select events in the database.
- Executable tools: External executables that provide additional functions to the event list (such as an open browser tool).

Tools can also be associated with a class of alert.

- Database tables and fields

All database tables can be modified by an administrator apart from system tables. Custom table and fields can be added using the administrative console or nco_sql interface.

- Database triggers

Automation is used to detect changes in the ObjectServer and execute automated responses to these changes. This enables the ObjectServer to process alerts without requiring an operator to take action. There is a set of default automations, but more can be added by an administrator to provide more advanced functionality and events management.

- Classes

Alerts in the ObjectServer have a class assigned by the probe. Each class can be associated with an event list tool menu that contains useful tools for alerts of that specific class.

► Desktop tools (event list)

The desktop is an integrated suite of graphical tools used to view and manage alerts and to configure how alert information is presented. Desktop tools are available on both UNIX and Windows platforms. Alert information is delivered in a format that allows users to quickly determine the availability of services on a network. When an alert cause has been identified, desktop tools enable users to resolve problems quickly.

The event list enables you to view and manage alerts. An alert is created when the ObjectServer receives an event, alarm, message, or data item. Each alert comprises fields of information held in a row of the ObjectServer alerts.status table.

Information about alerts is displayed in the event list according to filters and views:

- Filters enable you to display a subset of alerts based on specific criteria.
- Views enable you to choose which alert fields to display.

► Command-line SQL interface

The ObjectServer provides a Structured Query Language (SQL) interface for defining and manipulating relational database objects, such as tables and views. You can use the SQL interactive interface (called nco_sql on UNIX and isql on Windows) to connect to an ObjectServer and use SQL commands to interact with and control the ObjectServer. The SQL interactive interface enables you to perform tasks, such as creating a new database table or stopping the ObjectServer.

2.1.3 Groups, roles, and users

IBM Tivoli Netcool/OMNIBus security models use roles and allows an administrator an advanced level of control over user access and privileges:

- Roles: Roles are sets of permission assigned to a group and are used to determine the types of action users of that specific group can perform. Table 2-1 lists the default roles. A normal user would usually has the CatalogUser, AlertsUser, and DesktopAdmin roles.

Table 2-1 Roles

Role	Description
CatalogUser	This role includes permissions to view information about system, tools, security, and desktop database tables. This role provides a basis for IBM Tivoli Netcool/OMNIBus permissions. This role does not provide sufficient permissions to use any IBM Tivoli Netcool/OMNIBus applications. All groups should be assigned this role.

Role	Description
AlertsUser	This role includes permissions to view, update, and delete entries in the alerts.status table, view, insert, and delete entries in the alerts.journal table, and view and delete entries in the alerts.details table. This role, in combination with the CatalogUser role, enables you to display and manipulate alerts, create filters and views, and run standard tools in the event list.
AlertsProbe	This role includes permissions to insert and update entries in the alerts.status table and insert entries in the alerts.details table. This role, in combination with the CatalogUser role, provides the permissions that a probe needs to generate alerts in the ObjectServer. These permissions should be granted to any user that runs a probe application.
AlertsGateway	This role includes permissions to insert, update, and delete entries in the alerts.status, alerts.details, and alerts.journal tables, and the tables in the transfer database. The transfer database is used internally by the bi-directional ObjectServer Gateway to synchronize security information between ObjectServers. This role, in combination with the CatalogUser role, provides the permissions that a gateway needs to generate alerts in the ObjectServer. These permissions should be granted to any user that runs a gateway application.
DatabaseAdmin	This role includes permissions to create databases and files, and to create tables in the alerts, tools, and service databases. This role also includes permissions to modify or drop the alerts.status, alerts.details, and alerts.journal tables. This role, in combination with the CatalogUser role, provides permissions to create relational data structures in the ObjectServer.
AutoAdmin	This role includes permissions to create trigger groups, files, SQL and external procedures, and user signals. This role also includes permissions to create, modify, and drop triggers in the default trigger groups and modify or drop default trigger groups. This role, in combination with the CatalogUser role, provides permissions to create automations in the ObjectServer.
ToolsAdmin	This role includes permissions to delete, insert, and update all tools tables. This role, in combination with the CatalogUser role, provides permissions to create and modify tools that can be run from the desktop and IBM Tivoli Netcool/OMNIBus Administrator.
DesktopAdmin	This role includes permissions to update all desktop catalogs to insert, update, and delete colors, visuals, menus, classes, resolutions, and conversions. This role, in combination with the CatalogUser role, provides permissions to customize the desktop.
SecurityAdmin	This role, in combination with the CatalogUser role, includes permissions to manipulate users, groups, and roles using IBM Tivoli Netcool/OMNIBus Administrator or the SQL interactive interface. This role also includes permissions to set properties and drop user connections.
ISQL	This role, in combination with the CatalogUser role, includes permission to view ObjectServer data using the SQL interactive interface.

Role	Description
ISQLWrite	This role, in combination with the CatalogUser role, includes permission to view and modify ObjectServer data using the SQL interactive interface.
SuperUser	This role has all available permissions. You cannot modify the SuperUser role.
Public	All users are assigned this role. By default, the Public role is not assigned any permissions. You can modify, but not drop, the Public role.

- **Groups:** Groups are created in order to easily manage users belonging to the same group. A group is typically assigned one or more role as necessary. Table 2-2 lists the default groups.

Table 2-2 Groups

Group	Description
Probe	This group is assigned the CatalogUser, AlertsUser, and AlertsProbe roles.
Gateway	This group is assigned the CatalogUser, AlertsUser, and AlertsGateway roles.
ISQL	This group is assigned the ISQL role.
ISQLWrite	This group is assigned the ISQLWrite role.
Public	This group is assigned the Public role. All users are members of this group.
Normal	This group is assigned the CatalogUser and AlertsUser roles. This group cannot be deleted or renamed.
Administrator	This group is assigned the CatalogUser, AlertsUser, ToolsAdmin, and DesktopAdmin roles.
DesktopAdmin	This group cannot be deleted or renamed.
System	This group is assigned the CatalogUser, AlertsUser, ToolsAdmin, DesktopAdmin, AlertsProbe, AlertsGateway, DatabaseAdmin, AutoAdmin, SecurityAdmin, ISQL, ISQLWrite, and SuperUser roles. This group cannot be deleted or renamed.

- ▶ **Users:** A user represents an individual identity that can access IBM Tivoli Netcool/OMNIBus. Each user is connected as a member of a group to get access to the roles. There are some default users defined, as listed in Table 2-3.

Table 2-3 Users

User name	Description
root	This user is created with an empty string as a password by default. You can reset the password by using IBM Tivoli Netcool/OMNIBus Administrator, or use the ALTER USER ObjectServer SQL command.
nobody	This user is disabled and cannot be used to access the ObjectServer. Ownership of each alert in the alerts.status table is assigned to a user when the row is inserted. By default, probes assign rows to the nobody user.

- ▶ **Restriction filter:** A restriction filter is used to restrict the rows displayed on a user event list. Restriction filters need to be created and assigned to a user or group to control the data that can be displayed and modified from client and SQL commands.

2.1.4 Insert Delete Update Cycle (IDUC) components

IDUC is used to perform activity on the ObjectServer database. A large amount of data is passed through IBM Tivoli Netcool/OMNIBus components, such as:

- ▶ Between an ObjectServer and its client
- ▶ Between an ObjectServer and the gateways

To avoid the object server being overloaded with requests for event list (or gateway) updates, the ObjectServer sends a prompt to the desktop client and gateway every time an update is needed and then the desktop requests the update from the ObjectServer. This information is sent by way of a specific link named IDUC.

By default, a random port is chosen for the IDUC communications link. This port must be specified manually when connecting through a firewall in addition to the port on which the ObjectServer is running.

By default, the update interval is set to 60 seconds using the ObjectServer property named Granularity. This can be changed, and is usually reduced to improve the clients' response time, but keep in mind that it can greatly increase the network traffic and ObjectServer processing load.

2.2 IBM Tivoli Netcool/OMNibus configuration

There are several configurations that can be used for installing and configuring IBM Tivoli Netcool/OMNibus. This section discusses the following configurations:

- ▶ 2.2.1, “Without failover” on page 45
- ▶ 2.2.2, “With failover” on page 46
- ▶ 2.2.3, “Desktop ObjectServer” on page 47
- ▶ 2.2.4, “Tiered implementation” on page 48
- ▶ 2.2.5, “Probe features” on page 49

2.2.1 Without failover

With this simple architecture, probes, desktops, and gateways connect to a single ObjectServer (no backup server and no failover mode). If the ObjectServer fails, the desktop component will not work, while probes and gateways will use the store and forward mode. Figure 2-2 shows this configuration.

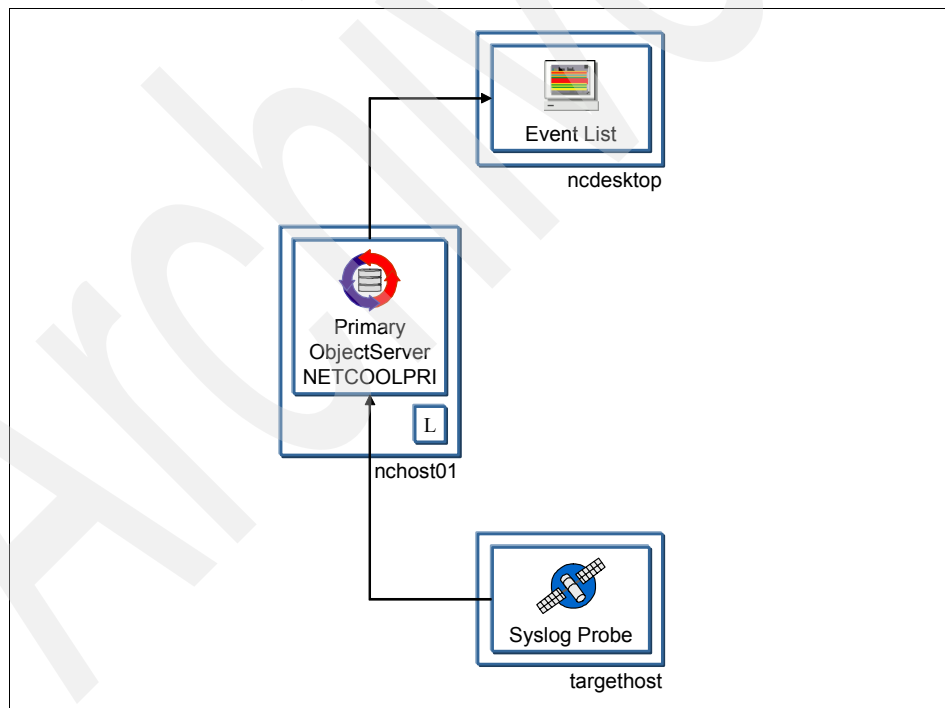


Figure 2-2 No failover

2.2.2 With failover

In a failover architecture, ObjectServers are configured as a virtual ObjectServer. In this case, desktops, gateways, and probes are connected to the failover pair and if the primary object server fails, they will switch to the backup object server automatically. This configuration also supports fallback. When the primary object server is available again, the probe and desktop will reconnect automatically to it. Figure 2-3 shows this configuration.

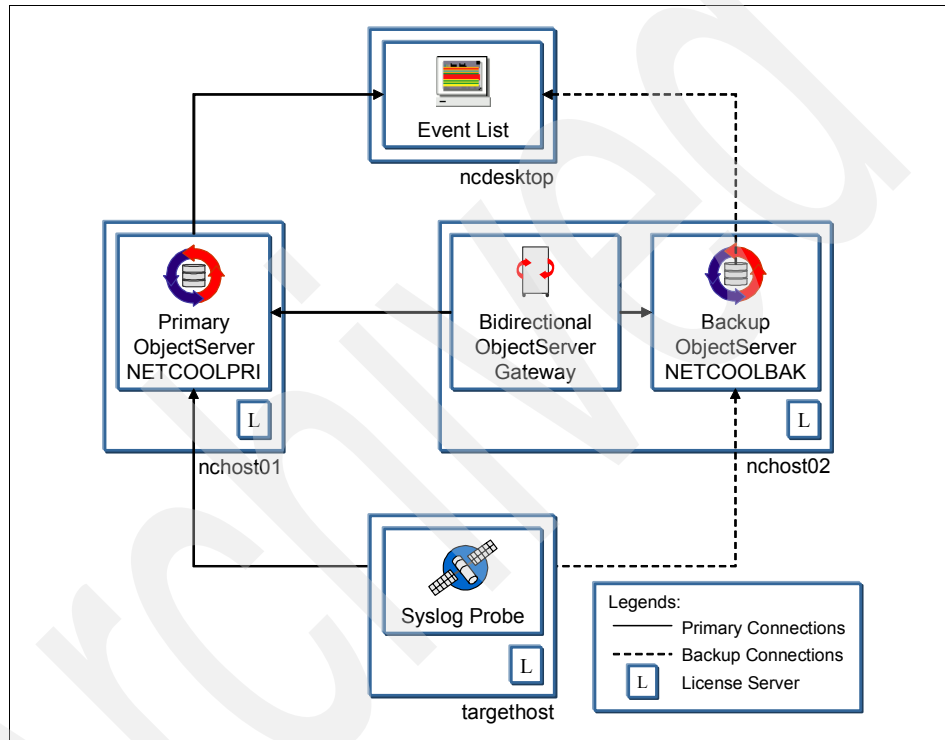


Figure 2-3 Failover architecture

The virtual ObjectServer is configured as a Primary and Backup pair to a single ObjectServer definition.

2.2.3 Desktop ObjectServer

In order to improve overall performance of the system, you can use another advanced architecture called Desktop ObjectServer. For example, if you have many distributed ObjectServers sending events to a central ObjectServer using a uni-directional gateway, and many desktop clients connecting to the central object server, you can experience frequent high loads on the server based on the number of clients connected and events inserted from the gateway. Using the Desktop ObjectServer architecture, you can:

- ▶ Minimize the load on the central Object Server
- ▶ Increase desktop clients' performance
- ▶ Improve latency
- ▶ Maintain high data integrity and consistency by simultaneously updating the central ObjectServer

This architecture is composed of one Master ObjectServer and one or more Desktop ObjectServers. Dual Server Desktops (DSDs) connect to a Master ObjectServer when performing writes, but connect to a Desktop ObjectServer when displaying data, as shown in Figure 2-4.

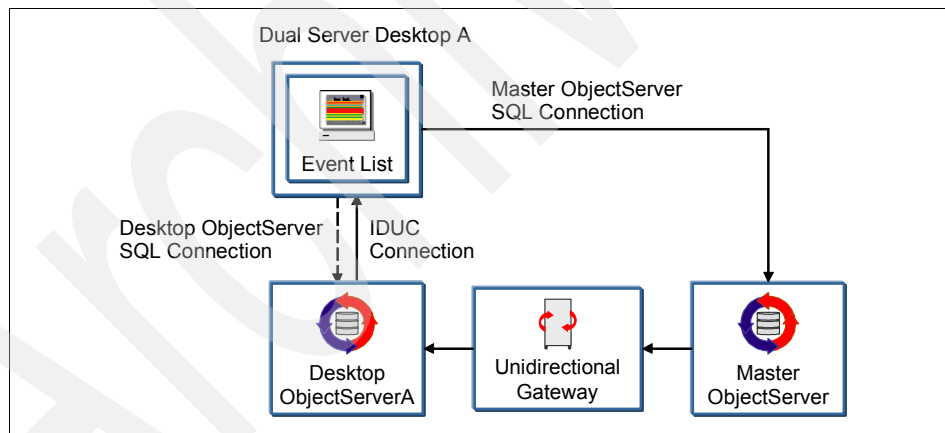


Figure 2-4 Desktop ObjectServer

This architecture is designed to help you avoid performance issues. It is easily scalable by adding more ObjectServers to accommodate new desktops and is easier to maintain by having different ObjectServers for different roles (a Collection ObjectServer collecting events and sending them to Master ObjectServer, and a Desktop ObjectServer for displaying data to users). It also reduces network utilization across WAN links.

2.2.4 Tiered implementation

A complex and large network and enterprise environment requires a more specialized implementation configuration. This kind of configuration is often referred as *tiered implementation*. In a tiered implementation, multiple ObjectServers are deployed in the environment, as shown in Figure 2-5.

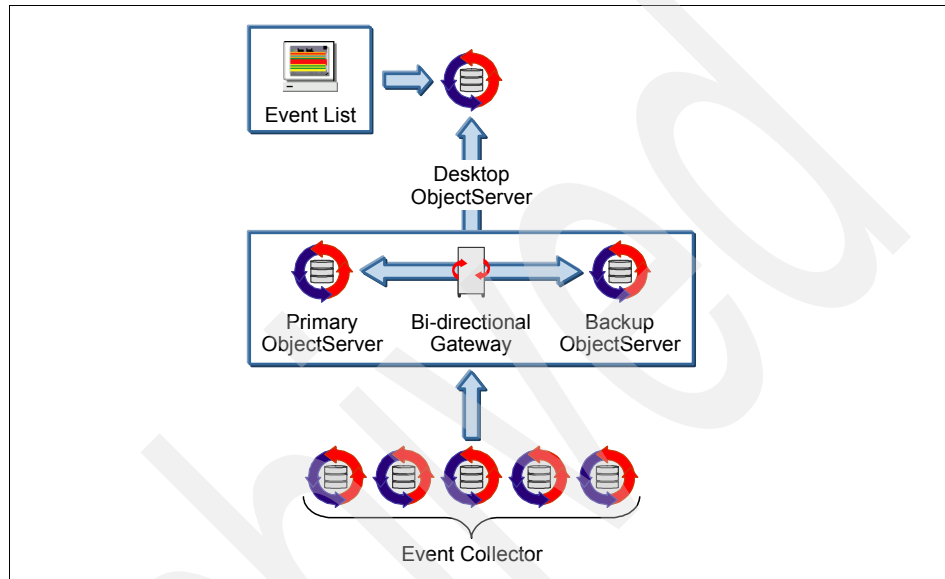


Figure 2-5 Tiered ObjectServer implementation

The tiers are:

- ▶ **Collector ObjectServers:** These ObjectServers are located on various regional locations of the enterprise and typically connected to the probes and monitors. Their primary function is to collect events. They perform basic deduplication and automation to ensure that events that are sent to the central ObjectServers are unique and relevant.
- ▶ **Central ObjectServers:** These ObjectServers collect events from various collectors through uni-directional gateway. These ObjectServers typically act as a redundant (failover) virtual ObjectServer and provide the main automation and event processing.
- ▶ **Desktop ObjectServers:** These ObjectServer retrieve events from the central ObjectServers for display on the individual cluster of users that monitor events.

2.2.5 Probe features

A probe has the following features:

- ▶ **Store and Forward Mode:** Probes can continue to run if the target ObjectServer is down. When the probe detects that the ObjectServer is not present (usually because it is unable to forward an alert to the ObjectServer), it switches to store mode. In this mode, the probe writes all of the messages it would normally send to the ObjectServer to a file named `$OMNIHOME/var/probename.servername.store`.
- ▶ **Automatic Store and Forward:** By default, store and forward mode is active only after a connection to the ObjectServer has been established, used, and then lost. If the ObjectServer is not running when the probe starts, store and forward mode is not triggered and the probe terminates unless you set the probe to run in automatic store and forward mode, it will go straight into store mode if the ObjectServer is not running, as long as the probe has been connected to the ObjectServer at least once before.
- ▶ **Raw Capture Mode:** Raw capture mode enables you to save the complete stream of event data acquired by a probe into a file without any processing by the rules file. This can be useful for auditing, recording, or debugging the operation of a probe.
- ▶ **Secure Mode:** You can run the ObjectServer in secure mode. This makes ObjectServer authenticate probe, gateway, and proxy server connections by requiring a user name and encrypted password.
- ▶ **Peer-to-Peer Failover:** Two instances of a probe can run simultaneously in a peer-to-peer failover relationship. One instance is designated as the master; the other acts as a subordinate and is on hot standby. If the master instance fails, the subordinate instance is activated.

2.3 Configuration planning issues

There are several configuration planning issues that must be resolved before the implementation to ensure a workable solution and that implementation is a success. The issues are:

- ▶ 2.3.1, “Naming convention” on page 50
- ▶ 2.3.2, “SSL usage” on page 50
- ▶ 2.3.3, “Port and firewall considerations” on page 52

2.3.1 Naming convention

While the default names of objects are supplied with the installation, such as NCOMS for ObjectServer, NCO_PA for process agent, and so on, these names are not adequate or descriptive enough for a large environment. A naming convention is necessary to quickly identify a component for quick problem determination and recovery.

There are several attributes that can be used for naming conventions, such as:

- ▶ Function: ObjectServer, Gateway, Probes, Process agent, and so on
- ▶ Position: Desktop, Collector, and Central
- ▶ Location: Region, country, state, and city
- ▶ Failover status: Primary or backup

2.3.2 SSL usage

SSL communications can be implemented between IBM Tivoli Netcool/OMNIBus components, such as probes and monitors to ObjectServers, and gateways to ObjectServers. However, configuring SSL communication between probes and monitors to the subsystem it monitors depend a lot on the subsystem. Some communication methods to the subsystems do not supporting SSL.

Local monitoring from probes and monitors do not typically use network communication, so it would not generate an unencrypted network packet. A typical implementation configuration with SSL is shown in Figure 2-6.

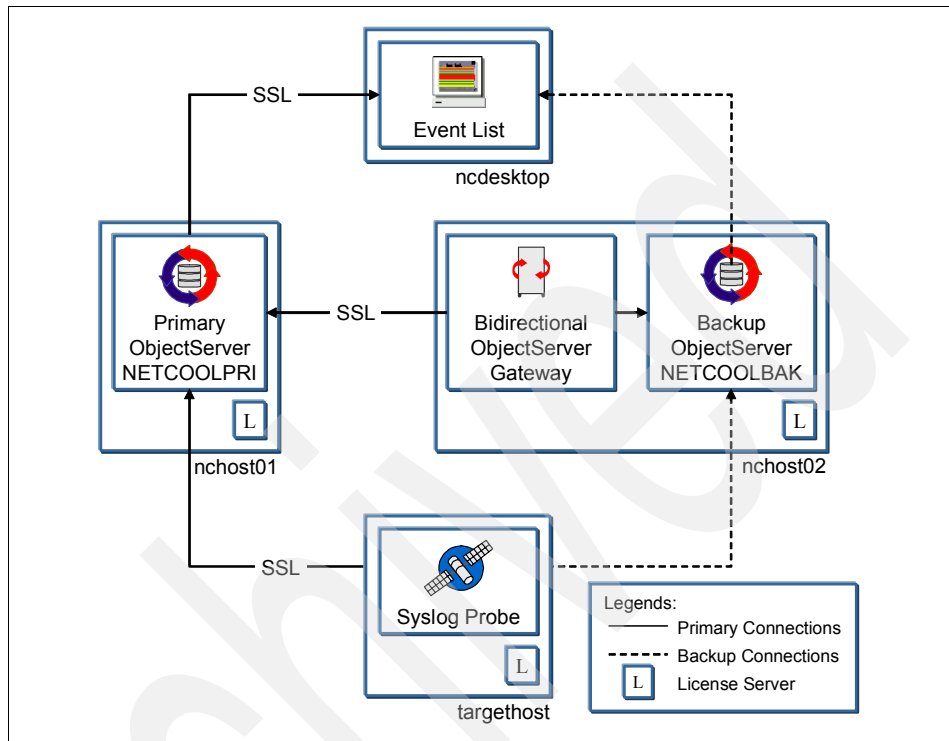


Figure 2-6 SSL communication

2.3.3 Port and firewall considerations

Typically, firewall concerns arise for communications between a gateway and ObjectServer and an event list client to an ObjectServer. The communication for a desktop event list is shown in Figure 2-7.

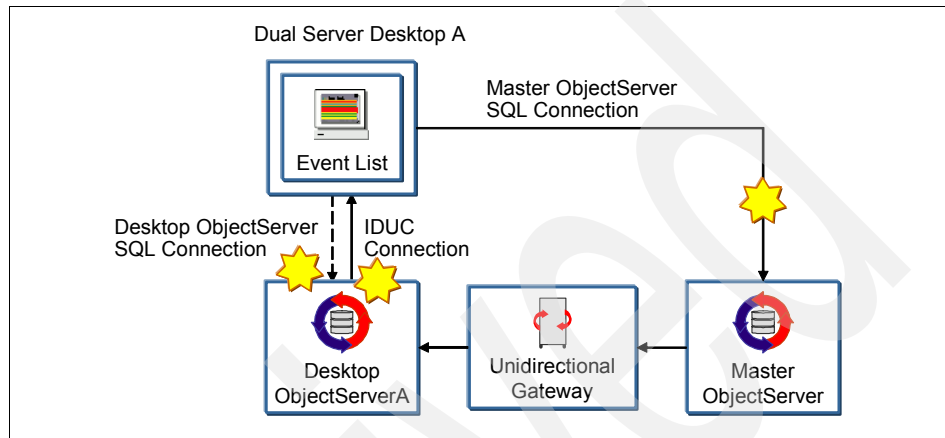


Figure 2-7 Desktop communication

In Figure 2-7, the desktop must open the IDUC port and the ObjectServer port on the desktop ObjectServer. It also opens a connection to the Master ObjectServer port.

Gateways communicate with the ObjectServers using the standard ObjectServer ports defined in the interface files.

2.4 ObjectServer tables

The ObjectServer is based on a relational database structure. Some of the important tables for the ObjectServer are listed in Table 2-4 on page 53. Understanding the tables and their usage are important in defining the procedures, triggers and SQL commands for processing events in ObjectServer.

Table 2-4 Table name

Table name	Usage
Master tables	
master.stats	The master.stats table stores timing information about the alerts.status, alerts.details, and alerts.journal tables. This timing information is gathered if the stats_triggers trigger group is enabled. The stats_triggers trigger group is disabled by default in the automation.sql file.
master.national	The master.national table identifies the master ObjectServer and the dual-write mode in a desktop ObjectServer architecture.
master.servergroups	The master.servergroups table is used to load-balance desktop connections.
Catalog tables	
catalog.memstores	The catalog.memstores table stores information about memstores, including the hard and soft limits of the memstore size, and how many bytes are currently being used.
catalog.databases	The catalog.databases table stores information about each database, including the number of objects in the database and the type of database (system or user).
catalog.tables	The catalog.tables table stores information about all types of tables, including system and user tables, views, and transition tables.
catalog.base_tables	The catalog.base_tables table stores information about user and system tables.
catalog.views	The catalog.views table stores information about views. The CreationText column contains the SQL used to create the view.
catalog.files	The catalog.files table stores information about ObjectServer files, including the path to the operating system file associated with each ObjectServer file.
catalog.restrictions	The catalog.restrictions table stores information about restriction filters. The ConditionText column contains the SQL condition.
catalog.columns	The catalog.columns table stores information about table columns, including their data types.
catalog.primitive_signals	The catalog.primitive_signals table stores information about user and system signals.
catalog.primitive_signal_parameters	The catalog.primitive_signal_parameters table stores information about the parameters to system and user defined signals.

Table name	Usage
catalog.trigger_groups	The catalog.trigger_groups table stores information about trigger groups, including whether or not the trigger group is enabled.
catalog.triggers	The catalog.triggers table stores information about triggers, including the type of trigger, the trigger priority, and what trigger group it is in.
catalog.database_triggers	These tables store information about triggers, including the type of operation that causes the trigger to fire.
catalog.signal_triggers	
catalog.temporal_triggers	
catalog.procedures	The catalog.procedures table stores information about procedures, including whether the procedure is an SQL procedure or an external procedure.
catalog.sql_procedures	The catalog.sql_procedures table stores information about SQL procedures, including the code for the procedure.
catalog.external_procedures	The catalog.external_procedures table stores information about external procedures, including the name of the procedure executable and the host on which it runs.
catalog.procedure_parameters	The catalog.procedure_parameters table stores information about procedure parameters, including parameter types.
catalog.connections	The catalog.connections table contains information about connections to the ObjectServer.
catalog.properties	The catalog.properties table contains information about ObjectServer properties.
catalog.security_permissions	The catalog.security_permissions table contains permission information for ObjectServer objects. This table is used only by the Netcool/OMNIbus administrator.
catalog.profiles	The catalog.profiles table contains timing information for the execution of SQL commands from client connections.
catalog.trigger_stats	The catalog.trigger_stats table stores timing information about triggers, including the number of times the trigger has been raised and the number of times the trigger has fired. These statistics are gathered unless the automation system is disabled by setting the -autoenabled command-line option to FALSE. Trigger statistics are also logged to the file \$NCHOME/omnibus/log/servername_trigger_stats.logn.

Table name	Usage
Alert related tables	
alerts.status	The alerts.status table contains status information about problems that have been detected by probes. Resolved entries are automatically removed by an automation program.
alerts.details	The alerts.details table contains the detail attributes of the alerts in the system. This is sent from the probes with a detail(\$*) statement.
alerts.journal	The alerts.journal table provides a history of work performed on alerts.
alerts.iduc_messages	The alerts.iduc_messages table is required to support internationalization (i18n) and is used to send i18n-safe Insert, Delete, Update, or Control (IDUC) client messages. This table ensures that characters transferred across varying encodings are converted into recognizable formats.
alerts.objclass	The alerts.objclass table is used to determine which menu and icons to use for a particular class of object.
alerts.objmenus	The alerts.objmenus table is used to provide the list of menus.
alerts.objmenuitems	The alerts.objmenuitems table is used to provide the content of menus.
alerts.resolutions	The alerts.resolutions table is used to maintain the Resolutions option in the event list.
alerts.conversions	The alerts.conversions table is used to provide easy conversion from a numeric value to a string for any column.
alerts.col_visuals	The alerts.col_visuals table is used to provide the default visuals for columns when displayed in the desktop tools.
alerts.colors	The alerts.colors table is used to create the colors required by the Windows desktop.
Tools related tables	
tools.actions	The tools.actions table is used to control desktop tools.
tools.action_access	The tools.action_access table is used to control access to desktop tools.
tools.menus	The tools.menus table is used to control desktop tool menus.
tools.menu_items	The tools.menu_items table is used to control desktop tool menu items.
tools.prompt_defs	The tools.prompt_defs table is used to store all prompt definitions.

Table name	Usage
tools.menu_defs	The tools.menu_defs table is used to control desktop tool menu items.
Service table	
service.status	The service.status table is used to control the additional features required to support Netcool Internet Service Monitoring.

The tables can be queried using the `nco_sq1` or `isq1` commands. Some useful SQL queries are:

- ▶ Finding the number of occurrences:
SELECT COUNT(*) FROM <tables>
- ▶ Finding the latest occurrences:
SELECT MAX(<col-name>) FROM <tables>

The WHERE selection clauses can also be critical for the performance of the SQL query. The WHERE clauses are used to filter returned rows, evaluate the conditions, and filter them sequentially. The order of most efficient queries are:

- ▶ Integer field matching
- ▶ Exact character matching
- ▶ Wild card character matching
- ▶ Regular expression character matching

One of the most important tables is the alerts.status table. This table stores all open alerts in IBM Tivoli Netcool/OMNIBus. The alerts.status table structure is listed in Table 2-5.

Table 2-5 The alerts.status table

Column name	Data type	Description
Identifier	varchar(255)	Controls ObjectServer deduplication.
Serial	incr	The Tivoli Netcool/OMNIBus serial number for the row.
Node	varchar(64)	Identifies the managed entity from which the alarm originated. This could be a host or device name, service name, or other entity.
NodeAlias	varchar(64)	The alias for the node. For network devices or hosts, this should be the logical (layer-3) address of the entity. For IP devices or hosts, this should be the IP address.
Manager	varchar(64)	The descriptive name of the probe that collected and forwarded the alarm to the ObjectServer. This can also be used to indicate the host on which the probe is running.

Column name	Data type	Description
Agent	varchar(64)	The descriptive name of the sub-manager that generated the alert.
AlertGroup	varchar(255)	The descriptive name of the type of failure indicated by the alert (for example, Interface Status or CPU Utilization).
AlertKey	varchar(255)	The descriptive key that indicates the managed object instance referenced by the alert (for example, the disk partition indicated by a file system full alert or the switch port indicated by a utilization alert).
Severity	integer	Indicates the alert severity level, which provides an indication of how the perceived capability of the managed object has been affected. The color of the alert in the event list is controlled by the severity value.
Summary	varchar(255)	The text summary of the cause of the alert.
StateChange	time	An automatically maintained ObjectServer time stamp of the last insertions or update of the alert from any source.
FirstOccurrence	time	The time in seconds (from midnight 1 Jan, 1970) when this alert was created or when polling started at the probe.
LastOccurrence	time	The time when this alert was last updated at the probe.
InternalLast	time	The time when this alert was last updated at the ObjectServer.
Poll	integer	The frequency of polling for this alert in seconds.
Type	integer	The type of alert, that is, 0: not set, 1: problem, 2:resolution, and so on.
Tally	integer	Automatically maintained count of the number of insertions and updates of the alert from any source. This count is affected by deduplication.
Class	integer	The alert class used to identify the probe or vendor from which the alert was generated. Controls the applicability of context-sensitive event list tools.
Grade	integer	Indicates the state of escalation for the alert, that is, 0: not escalated and 1: escalated.
Location	varchar(64)	Indicates the physical location of the device, host, or service for which the alert was generated.
OwnerUID	integer	The user identifier of the user who is assigned to handle this alert. The default is 65534, which is the identifier for the nobody user.

Column name	Data type	Description
OwnerGID	integer	The group identifier of the group that is assigned to handle this alert.
Acknowledged	integer	Indicates whether the alert has been acknowledged (0/1)
Flash	integer	Enables the option to make the event list flash.
EventId	varchar(255)	The event ID (for example, SNMPTRAP-link down). Multiple events can have the same event ID. This is populated by the probe rules file and used by Tivoli Network Manager IP Edition.
ExpireTime	integer	The number of seconds until the alert is cleared automatically. Used by the Expire automation.
ProcessReq	integer	Indicates whether the alert should be processed by Tivoli Network Manager IP Edition. This is populated by the probe rules file and used by Tivoli Network Manager IP Edition.
SuppressEscl	integer	Used to suppress or escalate the alert.
Customer	varchar(64)	The name of the customer affected by this alert.
Service	varchar(64)	The name of the service affected by this alert.
PhysicalSlot	integer	The slot number indicated by the alert.
PhysicalPort	integer	The port number indicated by the alert.
PhysicalCard	varchar(64)	The card name or description indicated by the alert.
TaskList	integer	Indicates whether a user has added the alert to the Task List. Operators can add alerts to the Task List from the event list.
NmosSerial	varchar(64)	The serial number of a suppressed alert. Populated by Tivoli Network Manager IP Edition.
NmosObjInst	integer	Populated by Tivoli Network Manager IP Edition during alert processing.
NmosCauseType	integer	The alert state, populated by Tivoli Network Manager IP Edition as an integer value, that is, 0: unknown, 1: root cause, or 2: symptom.
NmosDomainName	varchar(64)	The name of the Tivoli Network Manager IP Edition domain that is managing the event.
NmosEntityId	integer	A unique numerical ID that identifies the Tivoli Network Manager IP Edition topology entity with which the event is associated.

Column name	Data type	Description
NmosManagedStatus	integer	The managed status of the network entity for which the event was raised. Can apply to events from Tivoli Network Manager IP Edition and from any probe. You can use this column to filter out events from interfaces that are not considered relevant.
LocalNodeAlias	varchar(64)	The alias of the network entity indicated by the alert. For network devices or hosts, this is the logical (layer-3) address of the entity, or another logical address that enables direct communication with the device. For use in managed object instance identification.
LocalPriObj	varchar(255)	The primary object referenced by the alert. For use in managed object instance identification.
LocalSecObj	varchar(255)	The secondary object referenced by the alert. For use in managed object instance identification.
LocalRootObj	varchar(255)	An object that is equivalent to the primary object referenced in the alarm. For use in managed object instance identification.
RemoteNodeAlias	varchar(64)	The network address of the remote network entity. For use in managed object instance identification.
RemotePriObj	varchar(255)	The primary object of a remote network entity referenced by an alarm. For use in managed object instance identification.
RemoteSecObj	varchar(255)	The secondary object of a remote network entity referenced by an alarm. For use in managed object instance identification.
RemoteRootObj	varchar(255)	An object that is equivalent to the remote entity's primary object referenced in the alarm. For use in managed object instance identification.
X733EventType	integer	Indicates the alert type 0-10.
X733ProbableCause	integer	Indicates the probable cause of the alert.
X733SpecificProb	varchar(64)	Identifies additional information for the probable cause of the alert. Used by probe rules files to specify a set of identifiers for use in managed object instance identification.
X733CorrNotif	varchar(255)	A listing of all notifications with which this notification is correlated.
ServerName	varchar(64)	The name of the originating ObjectServer. Used by gateways to control propagation of alerts between ObjectServers.

Column name	Data type	Description
ServerSerial	integer	The serial number of the alert on the originating ObjectServer (if it did not originate on this ObjectServer). Used by gateways to control the propagation of alerts between ObjectServers.
URL	varchar(1024)	Optional URL that provides a link to additional information in the vendor's device or Enterprise Network Management System (ENMS).
MasterSerial	integer	Identifies the master ObjectServer if this alert is being processed in a desktop ObjectServer environment. This column is added when you run the database initialization utility nco_dbinit with the -desktopserver option. MasterSerial <i>must</i> be the last column in the alerts.status table if you are using a desktop ObjectServer environment.
ExtendedAttr	varchar(4096)	Holds name-value pairs (of Tivoli Enterprise Console extended attributes) or any other additional information for which no dedicated column exists in the alerts.status table. Use this column only through the nvp_get, nvp_set, and nvp_exists SQL functions.

Installation

This chapter discusses the installation of IBM Tivoli Netcool/OMNibus. The topics discussed are:

- ▶ 3.1, “Installation process” on page 62
- ▶ 3.2, “Simnet probe installation” on page 65
- ▶ 3.3, “Installing a FixPack” on page 66
- ▶ 3.4, “Configuration export and import” on page 66
- ▶ 3.5, “Post-implementation customization” on page 68

3.1 Installation process

This section discusses the installation of IBM Tivoli Netcool/OMNIBus V7.2 on a UNIX or Linux server. For a sample of the wizard in Windows, see 4.1.1, “Remote desktop installation” on page 86.

The installation path defaults are:

- ▶ UNIX: /opt/netcool
- ▶ Windows: C:\Program Files\IBM\Tivoli\Netcool

Perform the following steps:

1. Download the required IBM Tivoli Netcool/OMNIBus binaries package C876RML from the IBM Passport advantage site, found at the following address:

http://www-01.ibm.com/software/howtobuy/passportadvantage/pao_customers.htm

Additional support and fixes can be downloaded from the IBM Tivoli Netcool/OMNIBus support site, found at the following address:

<http://www-01.ibm.com/software/sysmgmt/products/support/F495033D98219V85-download.html>

2. Put the binaries in an appropriate temporary location and extract the installation package.
3. Run the installation program; in UNIX, it is called `INSTALL`, and in Windows, it is called `setup`. There are some options for invoking the installation, such as:

-console	Forcing the installation to use the console mode.
-errorlevel	The logging level: debug, info, or warning.
-silent	Runs the installation in silent mode. Requires the <code>-params</code> option, which defines the XML file that is used as input.
-nchome	The Netcool home directory for UNIX installation.
-help	Shows the options for the installation command.

4. The following steps describe the console based installation steps in a UNIX platform. First, invoke the script `./INSTALL -console`. If this is the first time you have done this installation for this platform, supply the Netcool home directory, as shown in Example 3-1 on page 63.

Example 3-1 Home directory

Please enter installation directory [/opt/netcool]:
1548 blocks
. . .

Product: Netcool/OMNIBus
Welcome to the Netcool text installer for Netcool/OMNIBus

To continue press [Enter]:

5. Accept the license agreement, as shown in Example 3-2.

Example 3-2 License agreement

International Program License Agreement

Part 1 - General Terms

BY DOWNLOADING, INSTALLING, COPYING, ACCESSING, OR USING THE PROGRAM YOU AGREE TO THE TERMS OF THIS AGREEMENT. IF YOU ARE ACCEPTING THESE TERMS ON BEHALF OF ANOTHER PERSON OR A COMPANY OR OTHER LEGAL ENTITY, YOU REPRESENT AND WARRANT THAT YOU HAVE FULL AUTHORITY TO BIND THAT PERSON, COMPANY, OR LEGAL ENTITY TO THESE TERMS. IF YOU DO NOT AGREE TO THESE TERMS,

- DO NOT DOWNLOAD, INSTALL, COPY, ACCESS, OR USE THE PROGRAM; AND
- PROMPTLY RETURN THE PROGRAM AND PROOF OF ENTITLEMENT TO THE PARTY FROM WHOM YOU ACQUIRED IT TO OBTAIN A REFUND OF THE AMOUNT YOU PAID. IF YOU DOWNLOADED THE PROGRAM, CONTACT THE PARTY FROM WHOM YOU ACQUIRED IT.

"IBM" is International Business Machines Corporation or one of its subsidiaries.

"License Information" ("LI") is a document that provides information

Do you agree to be bound by the terms of this license (Yes/No) []: **Yes**

6. Select the required features. Example 3-3 shows that we selected all the features.

Example 3-3 Installation feature selection

Product: Netcool/OMNIBus

SelectFeature

[I] 1) Desktop - Desktop GUI Applications
[I] 2) Gateways - ObjectServer Gateways
[I] 3) Process Control - Process control and remote execution support.
[I] 4) Servers - ObjectServer and Proxy Server
[I] 5) Confpack - Confpack configuration backup and transfer tool
[I] 6) Administrator - Administrator configuration GUI
[I] 7) AEN Client - Accelerated Event Notification Client
[I] 8) Local Help System - Local On-line Help System. To use Standalone mode on-line help or start an Infocentre server, this feature must be installed.

Select:

- 1-8) Toggle feature
- s) Select all features
- u) Unselect all features
- i) Install selected features
- n) Next page (properties configuration)
- q) Quit

Option [i]: i

7. When the installation process completes correctly, you receive the message shown in Example 3-4.

Example 3-4 Installation completed

Installing product: Netcool/OMNIBus.

Product Netcool/OMNIBus installed OK.

If there is a problem with the installation, refer to the `$NCHOME/log/install/ncisetup.log` file for more information.

3.2 Simnet probe installation

IBM Tivoli Netcool/OMNIBus is now installed, so now you can install probes. The probes are installed as patches. These probe patches are installed using the `$NCHOME/omnibus/install/nco_patch` command. You can download the Simnet probe from the PassPort Advantage Web site (<http://www.ibm.com/software/passportadvantage>) using the part number C1BY0EN. Perform the following tasks:

1. Unzip the installation image to a temporary directory. The installation of the probe is performed as a patch. The patch is packaged as a tar.Z file. Do not extract the tar.Z file.
2. Run the `nco_patch` command to install the simnet probe from the compressed patch tar-ball in temporary directory.
3. Enter yes to install the probe.
4. Wait until the installation is completed

Example 3-5 shows the installation process.

Example 3-5 Probe installation

```
[netcool@omnibus tmp]$ unzip C1BY0EN.zip
Archive:  C1BY0EN.zip
  inflating: omnibus-3.x-linux2x86-probe-nco-p-simnet-4_0.tar.Z
  inflating: omnibus-3.x-linux2x86-probe-nco-p-simnet-4_0.README
[netcool@omnibus tmp]$ $NCHOME/omnibus/install/nco_patch
omnibus-3.x-linux2x86-probe-nco-p-simnet-4_0.tar.Z

----- End of README -----
Are you sure you want to install this patch? (y/n)? [default: y]
Patch "probe-nco-p-simnet-4" is successfully installed.
```

The `nco_patch` command has some other options, such as:

-remove	Removes a patch installation (including a probe).
-print	Lists the installed patch IDs.

3.3 Installing a FixPack

A FixPack for IBM Tivoli Netcool/OMNIBus V7.2 can be downloaded from the IBM support site at the following address:

<http://www-01.ibm.com/software/sysmgmt/products/support/F495033D98219V85-download.html>

The FixPack is installed by running the **nci setup** command under the `$NCHOME/install` directory. The FixPack is installed using the `-install` option, which points to the FixPack distribution file. A sample installation of FixPack 7.2.0.3 is shown in Example 3-6.

Example 3-6 FixPack installation

```
[netcool@omnibus]$ /opt/netcool/install/ncisetup -install
7.2.0.3-TIV-NCOMNIBUS-linux2x86-FP0003.jar-silent
Installing product: Netcool/OMNIBus.
Product Netcool/OMNIBus installed OK.
```

3.4 Configuration export and import

The IBM Tivoli Netcool/OMNIBus configuration of an ObjectServer can be exported and then imported to another ObjectServer. This process is very efficient for quickly cloning ObjectServer settings. This process is performed by running the **nco_confpack** command under `$NCHOME/omnibus/bin/`.

The **nco_confpack** command can be run in the following modes:

- ▶ Create a list of exportable configuration items using the `-list` option, as shown in Example 3-7 on page 67. You can choose the appropriate configuration that you want to export from the output file.

Example 3-7 nco_confpack -list

```
[netcool@omnibus tmp]$ $NCHOME/omnibus/bin/nco_confpack -list -server
NCOMS -file /tmp/NCOMS_list -user root
[netcool@omnibus tmp]$ cat /tmp/NCOMS_list
ObjectServer      NCOMS      Auto      audit_config_create_tool
ObjectServer      NCOMS      Auto      audit_config_drop_class
ObjectServer      NCOMS      Auto      audit_config_drop_col_visual
ObjectServer      NCOMS      Auto      audit_config_drop_conv
ObjectServer      NCOMS      Auto      audit_config_drop_menu
ObjectServer      NCOMS      Auto      audit_config_drop_object
ObjectServer      NCOMS      Auto      audit_config_drop_prompt
ObjectServer      NCOMS      Auto      audit_config_drop_tool
```

- ▶ Export the configuration using the `-export` option, as shown in Example 3-8. The configuration items in the list files are the only configuration items that will be exported.

Example 3-8 nco_confpack -export

```
[netcool@omnibus tmp]$ $NCHOME/omnibus/bin/nco_confpack -export -file
/tmp/NCOMS_list -package /tmp/NCOMS_package -user root
```

- ▶ Import a configuration into another ObjectServer using the `-import` option, as shown in Example 3-9.

Example 3-9 nco_confpack -import

```
[netcool@omnibus ~]$ $NCHOME/omnibus/bin/nco_confpack -import -package
/tmp/NCOMS_package -server NCOMS2 -user root
*****
*                               W A R N I N G                               *
*                               *                                           *
* This action may overwrite configuration currently in your system. *
*                               *                                           *
* It is recommended that a backup is made of the current data *
* before importing new data. *
*                               *                                           *
*****
Do you want to continue (y/n) [N]? y
```

A related concept to this topic is the backup of an ObjectServer configuration. To back up an ObjectServer database, you can run the following SQL command from a `nco_sql` or `isql` interface:

```
alter system backup '<backup_path>';
```

3.5 Post-implementation customization

There are several post-implementation customizations that must be performed for IBM Tivoli Netcool/OMNIBus. They are:

- ▶ 3.5.1, “Environmental variables” on page 68
- ▶ 3.5.2, “Directory ownership” on page 69
- ▶ 3.5.4, “Communication configuration” on page 72
- ▶ 3.5.5, “ObjectServer properties” on page 76

3.5.1 Environmental variables

The user of IBM Tivoli Netcool/OMNIBus must have several predefined environment variables set. These variables are typically set on the user's profile. The easiest method is to modify the `/etc/profile` file so that all users on the system will have those variables set. A sample modification of `/etc/profile` is shown in Example 3-10.

Example 3-10 Environmental variables for IBM Tivoli Netcool/OMNIBus

```
NCHOME=/opt/netcool/  
OMNIHOME=/opt/netcool/omnibus/  
LANG=C  
PATH=$PATH:/opt/netcool/omnibus/bin:/opt/netcool/netcool/install:/opt/n  
etcool/bin/  
LD_LIBRARY_PATH=/opt/netcool/platform/linux2x86/lib/  
  
export NCHOME OMNIHOME LANG PATH LD_LIBRARY_PATH
```

The variables to be set are:

NCHOME	Home directory of the Netcool installation
OMNIHOME	Home directory of IBM Tivoli Netcool/OMNIBus
LANG	Language convention
PATH	Executable search path
LD_LIBRARY_PATH	Dynamic link library loading path in Linux (LIBPATH is used in AIX®)

3.5.2 Directory ownership

On a UNIX/Linux operating system with an ObjectServer installed by the root user, the root user owns the \$NCHOME directory. A non-root user is typically used to own and run IBM Tivoli Netcool/OMNIBus. A typical setup is assigning the Netcool user as a member of the ncoadmin group, acting as the primary administrator with full access to the product functionality. The group can be defined with a specific group ID. Group IDs are defined in /etc/group and user IDs are defined in /etc/passwd.

If the group and user can be created, then you should add the root user to the group so it has access to it. Then you should change the file ownership of the \$NCHOME path. The commands to perform this task are shown in Example 3-11.

Example 3-11 Group creation on UNIX

```
[root@localhost ~]# groupadd -g 500 ncoadmin
[root@localhost ~]# useradd -g ncoadmin netcool
[root@localhost ~]# usermod -g ncoadmin root
```

3.5.3 ObjectServer creation and verification

Instances of ObjectServer can be created after the installation. To create the ObjectServer tables and information, use the **nco_dbinit** command, which resides in the \$NCHOME/omnibus/bin/ directory. To create an ObjectServer called NCOMS, issue this command:

```
$NCHOME/omnibus/bin/nco_dbinit -server NCOMS
```

The options for the ObjectServer can be specified as command-line options or in the property file \$NCHOME/omnibus/etc/nco_dbinit.props. The command executes the following SQL files:

application.sql	This file creates the default tables for the alerts and tools databases.
automation.sql	This file creates the default trigger groups and triggers.
desktop.sql	This file specifies default values for the desktop tables, including default colors, conversions, tools, and menus.
system.sql	This file specifies the security database and tables, and system users, groups, roles, and permissions. You must not edit this file.
security.sql	This file specifies additional operator and administrator roles.

For a Windows system, you can set up the ObjectServer to start as a Windows service using the **nco_objserv** command from the path %NCHOME%\omnibus\bin. To install the service, it uses the following options:

/install	Install a service.
/remove	Remove a service.
/instance	Define an optional instance name. The default service name is NCOObjectServer. The instance name is appended after a \$ sign after NCOObjectServer
/cmdline	Define the command line for the service that includes the name of the ObjectServer.
/noauto	When this option is set, the service will not autostart.

To create a service named NCOObjectServer\$ABC for an ObjectServer named MyABC, run this command:

```
nco_objserv /install /instance ABC /cmdline "-name MyABC"
```

Starting the ObjectServer manually can be done by running the **nco_objserv -name** command:

```
$NCHOME/omnibus/bin/nco_objserv -name <ObjectServer name>
```

You can then verify that the ObjectServer is running by using the **nco_ping** command, as shown in Example 3-12.

Example 3-12 IBM Tivoli Netcool/OMNIBus availability test on UNIX

```
[netcool@omnibus ~]$ nco_ping NCOMS  
NCO_PING: Server available.
```

You can then use the event list application locally using the **nco_event** command under \$NCHOME/omnibus/bin. The initial prompt is for the user name and password. The default user name is root with a blank password. The login window is shown in Figure 3-1 on page 71.

Note: As this is an X Window System application, the DISPLAY environment variable has to be set properly and access is granted from root using the `xhost` command.



Figure 3-1 IBM Tivoli Netcool/OMNibus event list login window on UNIX

Once the user name and password is verified, you are connected to the ObjectServer. There is a limit on the number of event lists that can connect to an ObjectServer. Figure 3-2 shows the event groups.

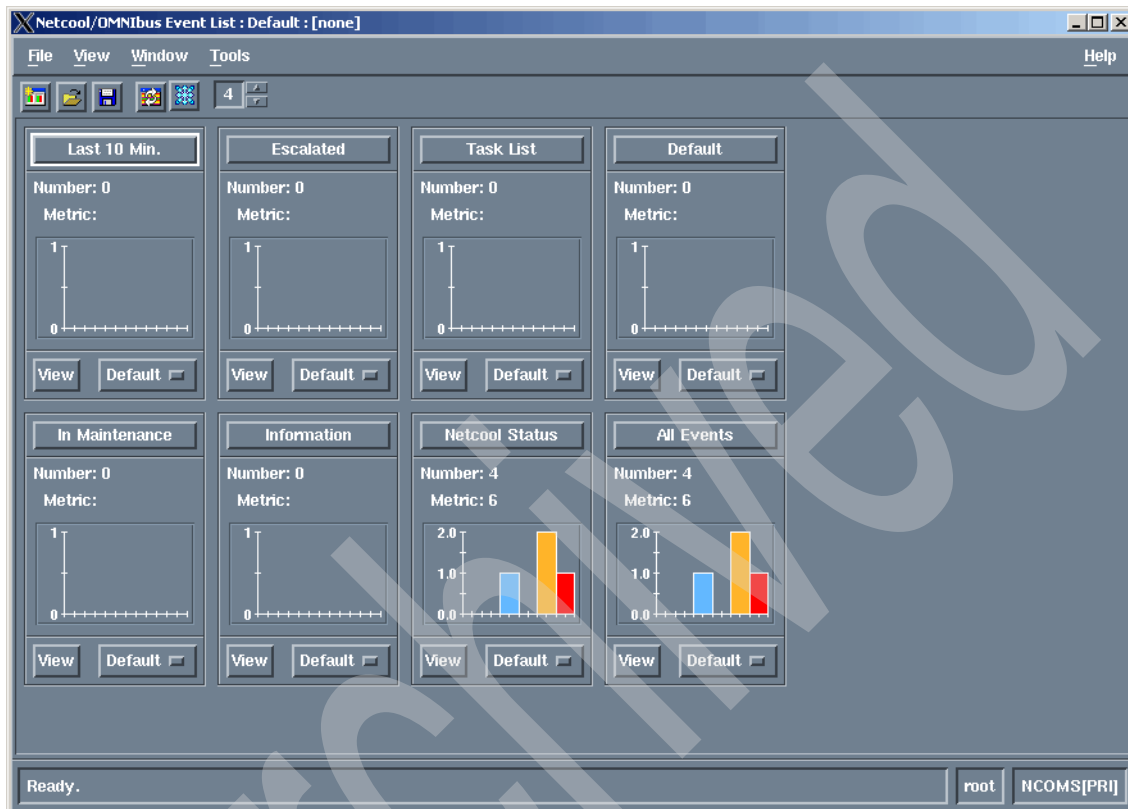


Figure 3-2 IBM Tivoli Netcool/OMNIBus event list window on UNIX

Each of the boxes in Figure 3-2 represents different filters of the event list that are available in IBM Tivoli Netcool/OMNIBus. The drop-down menus represent the view selection for the events that are retrieved from the particular filter.

3.5.4 Communication configuration

Communication between components are set in the interface file. The setting is different for UNIX or Linux and for Windows systems.

On UNIX or Linux

The communication configuration for a UNIX or Linux server includes changing the interface file. If the host name is resolvable to an IP address, edit the interface definition file `$NCHOME/etc/omni.dat` and replace `omnihost` with local servers host name. If the host name is not resolvable to an IP address, replace `omnihost` with the appropriate IPv4 or IPv6 address and port. Ensure that the port you define is not used from the output of the `netstat -an` command. Our definition file is shown in Example 3-13. Once you update the file, you can run the `nco_igen` command under `$NCHOME/bin` to redefine the interface file. The definition is for two separate ObjectServers.

Example 3-13 Interface definitions file for IBM Tivoli Netcool/OMNibus

```
[NCOMS]
{
    Primary: 9.42.170.132 4100
}
[NCOMS2]
{
    Primary: 9.42.170.132 4200
}
```

To define the above ObjectServers as a failover pair (virtual ObjectServer), define it as shown in Example 3-14.

Example 3-14 Interface definition file for virtual ObjectServer

```
[NCOMS]
{
    Primary: 9.42.170.132 4100
    Backup: 9.42.170.132 4200
}
```

Alternatively you can use the graphical interface (run `nco_xigen` to open it) to perform this update, as shown in Figure 3-3.



Figure 3-3 The `nco_xigen` window

On Windows

On a Windows platform, you set the communication configuration in the Netcool Server Editor. Select **All Programs** → **Netcool OMNIBUS** → **System Utilities** → **Server Editor**. The window shown in Figure 3-4 opens.

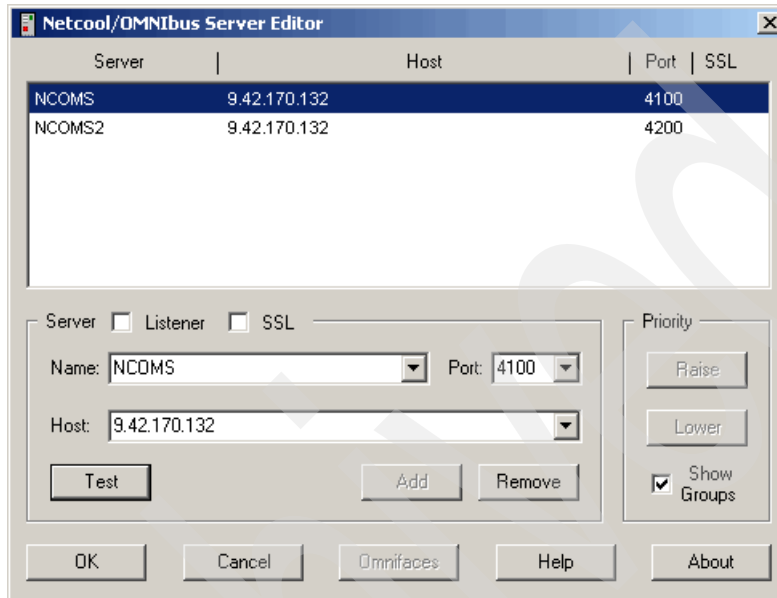


Figure 3-4 Interface definitions tool for IBM Tivoli Netcool/OMNIBus on Windows

You can define and test the ObjectServer. If you do, you should get the message “Server available”, as shown in Figure 3-5.

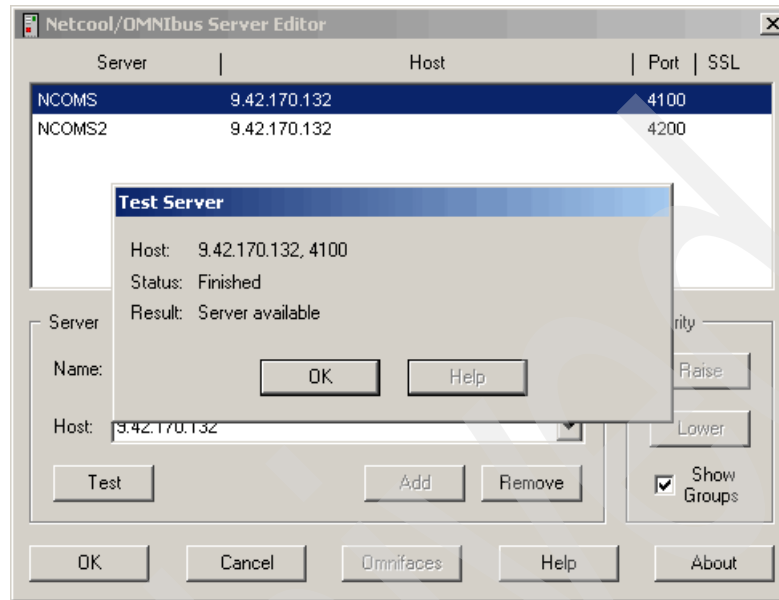


Figure 3-5 IBM Tivoli Netcool/OMNIBus availability test on Windows

3.5.5 ObjectServer properties

ObjectServer properties are stored in the file `$NCHOME/omnibus/etc/<servername>.props`. You can change the settings for the ObjectServer properties in the SQL interactive interface by running the `ALTER SYSTEM SET SQL` command, or by editing the properties file and restarting ObjectServer.

You can query the catalog.properties table to view information about the ObjectServer properties. The available properties are listed in Table 3-1.

Table 3-1 ObjectServer properties

Property	Description
AlertSecurityModel integer	This property determines whether group row level security is enforced in the event list. By default, group row level security is disabled (0). In this case: A member of the Normal group can modify a row that is assigned to themselves or the nobody user. A member of the Administrator group can modify a row that is assigned to themselves, the nobody user, or a member of the Normal group. If the AlertSecurityModel property is enabled (1), only users in the group that owns the row can modify the row. In this case, a member of the Normal or Administrator group can modify a row that is assigned to a group of which they are a member. A member of the System group can always modify any row.
AllowConnections TRUE FALSE	Specifies whether non-root users can connect to the ObjectServer. If FALSE, no new connections to the ObjectServer are allowed. The default is TRUE.
AllowISQL TRUE FALSE	Specifies whether connections to the ObjectServer are allowed using the SQL interactive interface. If FALSE, no user can connect using <code>nco_sq1</code> . The default is TRUE. If TRUE, this can be enabled for each user using the Netcool/OMNIBus Administrator.
AllowISQLWrite TRUE FALSE	Specifies whether modifications to the ObjectServer are allowed using the SQL interactive interface. If FALSE, no user can modify the ObjectServer using <code>nco_sq1</code> . The default is TRUE. If TRUE, this can be enabled for each user using the Netcool/OMNIBus Administrator.
AllowTimedRefresh TRUE FALSE	This property determines whether the user can enable a timed refresh in the Refresh tab of the Event List Preferences window. If TRUE, the event list preferences can be set to allow alert information to be updated at a specified interval rather than waiting for notification of updates from the ObjectServer. The default is FALSE. If FALSE, the timed refresh check box is grayed out in the Refresh tab of the Event List Preferences window and timed refresh is disabled.
Auto.Debug TRUE FALSE	If TRUE, automation debugging is enabled. The default is FALSE.

Property	Description
Auto.Enabled TRUE FALSE	If TRUE, automations are enabled. The default is TRUE.
Auto.StatsInterval integer	Specifies the interval in seconds at which the automation system collects statistics. The default is 60. Statistics are gathered unless the -autoenabled command-line option is set to FALSE, which disables all automations.
Auto.StatsLogfile string	Specifies the log file to which the automation system writes statistics. By default, the trigger statistics are logged to the file \$NCHOME/omnibus/log/servername_trigger_stats.logn.
BackupObjectServer TRUE FALSE	Provides failback capability with desktop clients, probes, the proxy server, and the ObjectServer Gateway. The default is FALSE; the desktop clients, probes, the proxy server, and gateways are assumed to be connected to a primary ObjectServer. When TRUE, the desktop clients, probes, the proxy server, and gateways are made aware that they are connected to the backup ObjectServer in a failover pair. If this is the case, the desktop clients, probes, the proxy server, and gateways will automatically check for the recovery of the primary ObjectServer in the failover pair and switch back (fail back) when it has restarted.
ClientHeartbeatDisable TRUE FALSE	Disables the client heartbeating system if set to TRUE. This causes a connected client to time out if the ObjectServer is busy, for example, during a gateway resynchronization or an automation. The default FALSE setting enables heartbeating, and prevents invalid and unnecessary client timeouts. If the ObjectServer is active but busy, this setting causes the ObjectServer to send a message to a connected client, with details of the type of processing in progress.
ClientHeartbeatRate integer	Sets the rate in seconds of a client heartbeat. This rate defines how long a client should wait for a response from the ObjectServer before timing out. The default value is 10.
Connections integer	Sets the maximum number of available connections for desktop clients, probes, and gateways. The default value is 30. Up to two connections may be used by the system.
DeleteLogFile string	The path and name of the delete log file, where all delete commands are recorded if delete logging is enabled. By default, deletes are logged to the file \$NCHOME/omnibus/log/servername_deletes_file.logn.
DeleteLogging TRUE FALSE	When TRUE, delete logging is enabled. The default is FALSE.

Property	Description
DeleteLogLevel integer	<p>The log level determines how much information is sent to the delete log file. Possible settings are:</p> <ul style="list-style-type: none"> ▶ <0: No logging. ▶ 0: Client type (application ID; for example, ctisql for nco_sql) and SQL executed. This is the default log level. ▶ 1: Time, user ID, client type, and SQL executed.
DeleteLogSize integer	<p>The maximum size of the delete log file. When the log file <code>servername_deletes_file.log1</code> reaches the specified size, it is renamed <code>servername_deletes_file.log2</code> and a new log file, <code>servername_deletes_file.log1</code>, is created. When the new file reaches its maximum size, the older file is deleted and the process repeats. The output from a single delete command is never split between log files. Therefore, log files may be larger than the specified size. The default log file size is 1024 KB.</p>
DTMaxTopRows integer	<p>The maximum number of rows that an administrator can specify when using the View Builder to restrict the number of rows an event list user can view. The default is 100.</p>
GWDeduplication integer	<p>This property controls the behavior when there is an attempt to reinsert an existing event in the ObjectServer. Possible values are:</p> <ul style="list-style-type: none"> ▶ 0: When set to this value (the default): <ul style="list-style-type: none"> – If the client is not a gateway, the Tally field is incremented. – The LastOccurrence, Summary, and AlertKey fields are updated with the new values and the StateChange and InternalLast fields are updated with the current time. – If the new Severity is greater than 0 (clear) and the old Severity was 0, the alert is also updated with the new Severity value. ▶ 1: If the client is a gateway, the new event replaces the old event. ▶ 2: If the client is a gateway, the new event is ignored. ▶ 3: When set to this value: <ul style="list-style-type: none"> – For all clients, the Tally field is incremented. – The LastOccurrence, Summary, and AlertKey fields are updated with the new values and the StateChange and InternalLast fields are updated with the current time. – If the new Severity is greater than 0 (clear) and the old Severity was 0, the alert is also updated with the new Severity value.

Property	Description
Granularity integer	Controls the update interval, in seconds, of IDUC broadcasts to desktops and gateways. Reducing this value increases the update rate to the clients. The default interval is 60 seconds.
Iduc.ListeningHostname string	Sets a host name for clients to use to locate the ObjectServer. On systems where DNS is used, the name returned by a host machine internally may not be the name by which it is referred to in the network. For example, a machine named "sfo" may actually be identified in the network as sfo.bigcorp.org. To allow clients to connect to the ObjectServer on sfo, set this property to sfo.bigcorp.org. The default is the name of the local machine, as reported by the hostname command.
Iduc.ListeningPort integer	Sets the port for the IDUC communication connection. This is the port on which the ObjectServer sends updates to each event list and gateway. If not specified, the IDUC port is selected by the ObjectServer at random from the unused port numbers available. The default is 0, that is, take the next available port. You can also specify the port in the /etc/services file on the host machine.
lpc.SSLCertificate string	Specifies the path to the SSL certificate. The default is \$NCHOME/etc/servername.crt. For more information about setting up a Netcool system using SSL communications, refer to the <i>IBM Tivoli Netcool/OMNIBus Installation and Deployment Guide</i> , SC23-6370.
lpc.SSLEnable TRUE FALSE	Enables SSL communications. For more information about setting up a Netcool system using SSL communications, refer to the <i>IBM Tivoli Netcool/OMNIBus Installation and Deployment Guide</i> , SC23-6370.
lpc.SSLPrivateKeyPassword string	Specifies the SSL private key password. The default is ". For more information about setting up a Netcool system using SSL communications, refer to the <i>IBM Tivoli Netcool/OMNIBus Installation and Deployment Guide</i> , SC23-6370.
MaxLogFileSize integer	Specifies the maximum size the log file can grow to, in kilobytes. The default is 1024 KB. Once it reaches the size specified, the servername.log file is renamed servername.log_OLD and a new log file is started. When the new file reaches the maximum size, it is renamed and the process starts again.

Property	Description
MessageLevel string	Specifies the message logging level. Possible values are debug, info, warn, error, and fatal. The default level is warn.
MessageLog string	Specifies the path to which messages are logged. The default is \$NCHOME/omnibus/log/NCOMS.log. On Windows, if the system cannot write to the specified log file (for example, as the result of a fatal error), it writes the error to a file named %NCHOME%\omnibus\log\err. The ObjectServer must be running as a service; otherwise, it cannot write errors to a file.
PA.Name string	Sets the process control agent name. This name must consist of 11 or fewer uppercase letters. When an external procedure is run from an automation, the ObjectServer can start an external process. To start the process, the ObjectServer contacts a process control agent. The default name for the process control agent is NCO_PA. This option is supported on both UNIX and Windows platforms.
PA.Password string	Specifies the password for the user connecting to a process control agent to run external procedures in automations. The default is ". If running external procedures in Netcool/OMNIBUS V7.x, the process control daemon must be able to authenticate the user. Therefore, a valid combination, which can be authenticated by the daemon, must be specified in the string values for PA.Username and PA.Password. Otherwise, the connection to the process control agent, as well as the execution of the external procedure, will fail. This option is only supported on UNIX platforms.
PA.Username string	Specifies the user name for connecting to a process control agent to run external procedures in automations. A value must be specified when the process control agent is running in secure mode. The default is root. Any user who requires access to the process control system must be a member of a UNIX user group (the default is ncoadmin) that you identify as an administrative group for this purpose. This option is only supported on UNIX platforms.
ProfileStatsInterval integer	Specifies the interval in seconds at which the profiler writes information to the profile log file. The default is 60 seconds.

Property	Description
Profile TRUE FALSE	Controls ObjectServer profiling. If TRUE, the amount of time it takes for clients to execute SQL is logged to the catalog.profiles table. The default is FALSE. The profile statistics are also logged to the file \$NCHOME/omnibus/log/ servername_profiler_report.logn file every profile statistics interval.
Props.CheckNames TRUE FALSE	When TRUE, the ObjectServer does not run if any specified property is invalid. The default is TRUE.
RegexpLibrary string	Defines which regular expression library to use for the ObjectServer. Possible values are NETCOOL and TRE. The default value of NETCOOL is useful for single-byte character processing and is recommended for optimal system performance. To enable the use of the extended regular expression syntax on single-byte and multi-byte character languages, set the value to TRE. Note that this setting results in decreased system performance.
RestrictionUpdateCheck TRUE FALSE	When FALSE, users with restriction filters applied can update alerts that will not appear in their view after the update. The default is TRUE.
RestrictPasswords TRUE FALSE	When TRUE, passwords must conform to the following restrictions: <ul style="list-style-type: none"> ▶ The password must consist of at least eight characters. ▶ The password must contain at least one numeric character. ▶ The password must contain at least one alphabetic character. The default is FALSE.
RestrictProxySQL TRUE FALSE	When TRUE, connections from a proxy server are restricted by the ObjectServer SQL commands that can be executed. The only modifications to ObjectServer data that can be made are inserts into the alerts.status and alerts.details tables. The default is FALSE. If FALSE, connections from a proxy server can execute any ObjectServer SQL commands.
Sec.AuditLevel string	Specifies the level of security auditing performed. Possible values are debug, info, warn, and error. The default is warn. The debug and info levels generate messages for authentication successes and failures, while warn and error levels generate messages for authentication failures only.
Sec.AuditLog string	Specifies the file to which audit information is written. The default is \$NCHOME/omnibus/log/servername/ audit.log.

Property	Description
Sec.UsePam TRUE FALSE	If TRUE and enabled for the user, external authorization can be used. The default is TRUE. For more information about PAM, refer to the <i>IBM Tivoli Netcool/OMNIBus Installation and Deployment Guide</i> , SC23-6370.
SecureMode TRUE FALSE	Sets the security mode of the ObjectServer. If TRUE, the ObjectServer authenticates probe, gateway, and proxy server connection requests with a user name and encrypted password. Other client connection requests are always authenticated with a user name and password.
Store.LocalizedSort TRUE FALSE	Defines whether localized sorting is enabled or disabled. The default is FALSE, which disables localized sorting in favor of standard C library (libc) string comparisons. For example, Å will be treated as a variant of A and the two characters will sort near each other. Use this default setting for optimal system performance. Set the value to TRUE to enable localized sorting. For example, in a Danish locale, Å will be treated as a separate letter that sorts just after Z. Note that specifying this setting results in decreased system performance.
UniqueLog TRUE FALSE	If TRUE, the log file is uniquely named by appending the process ID of the ObjectServer to the default log file name. For example, if the NCOMS ObjectServer is running as process 1234, the log file is named \$NCHOME/omnibus/log/NCOMS.1234.log. If the MessageLog property is set to sttderr or stdout, the UniqueLog property is ignored.

Archived

Configuration

This chapter discusses the configuration of IBM Tivoli Netcool/OMNIBus. The topics discussed are:

- ▶ 4.1, “Components configuration” on page 86
- ▶ 4.2, “Security configuration” on page 106
- ▶ 4.3, “ObjectServer customization” on page 117
- ▶ 4.4, “Probe configuration” on page 144
- ▶ 4.5, “ObjectServer to ObjectServer communication” on page 154
- ▶ 4.6, “Accelerated Event Notification client” on page 159
- ▶ 4.7, “IBM Tivoli Health Monitoring agent for ObjectServer V7.2” on page 171

4.1 Components configuration

This section discusses the configuration of the following components:

- ▶ 4.1.1, “Remote desktop installation” on page 86
- ▶ 4.1.2, “Gateway configuration” on page 91
- ▶ 4.1.3, “Process agent configuration” on page 100
- ▶ 4.1.4, “Startup script” on page 104

4.1.1 Remote desktop installation

When the operators want to have access to an IBM Tivoli Netcool/OMNIBus event list from their workstation, you need to install a remote desktop application to meet their wishes. This procedure is performed by using the standard IBM Tivoli Netcool/OMNIBus installation image, but you only need to install the desktop component. The following steps shows the Windows installation wizard for this installation:

1. Unzip and run the IBM Tivoli Netcool/OMNIBus install package named setup.exe. The window shown in Figure 4-1 opens. Click **Next**.



Figure 4-1 IBM Tivoli Netcool/OMNIBus installation

2. Accept the license agreement shown in Figure 4-2 on page 87. Click **Next**.

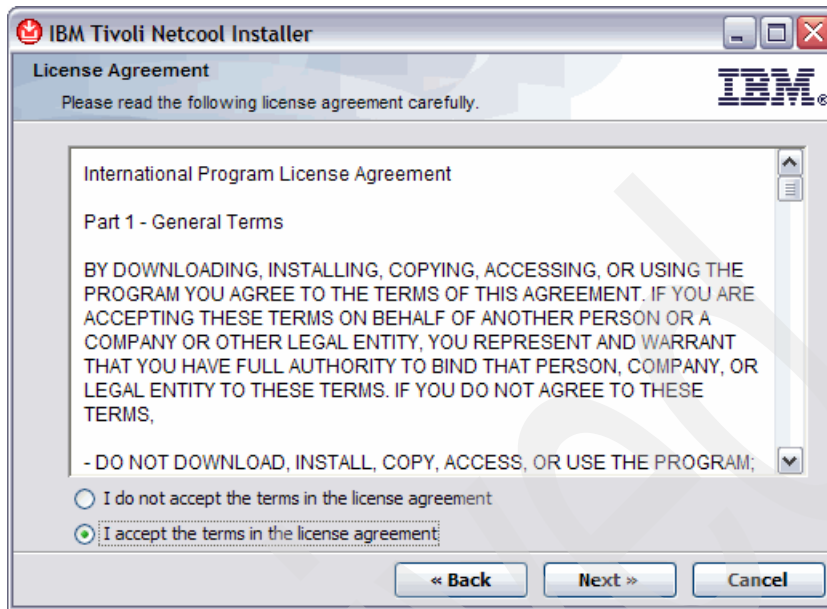


Figure 4-2 IBM Tivoli Netcool/OMNibus installation License Agreement

3. Choose the installation directory, as shown in Figure 4-3. Click **Next**.

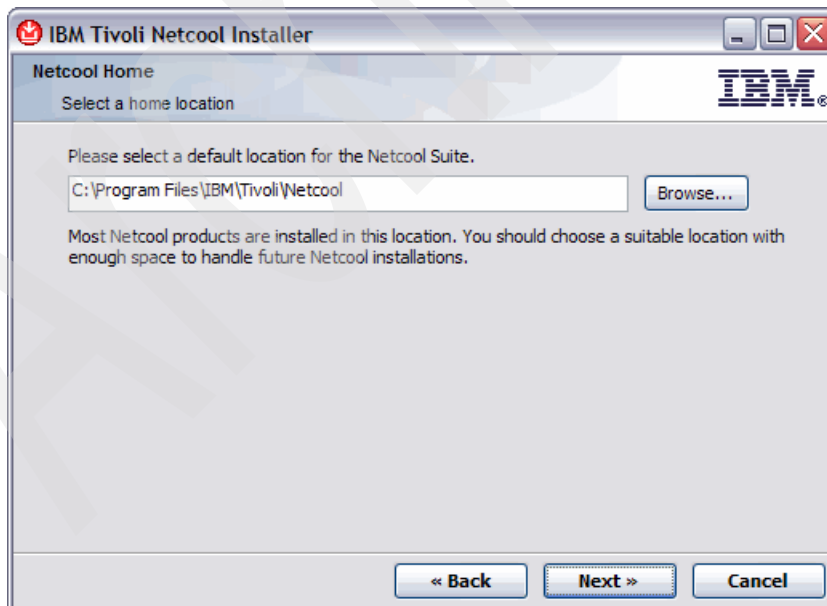


Figure 4-3 IBM Tivoli Netcool/OMNibus installation home location

4. Select the necessary components. For a remote desktop, you only select the Desktop component, as shown in Figure 4-4. Click **Next**.

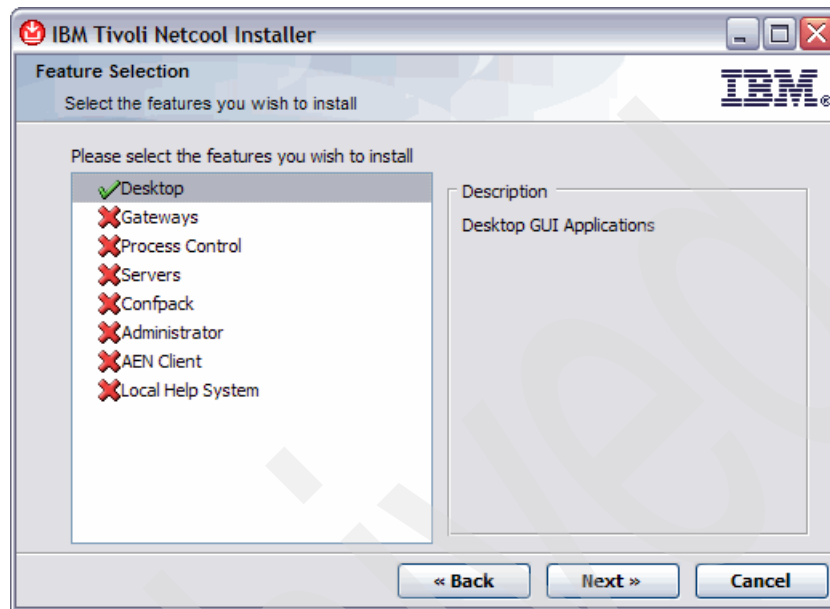


Figure 4-4 IBM Tivoli Netcool/OMNibus features selection

5. In the installation summary window shown in Figure 4-5 on page 89, click the **Install** button.

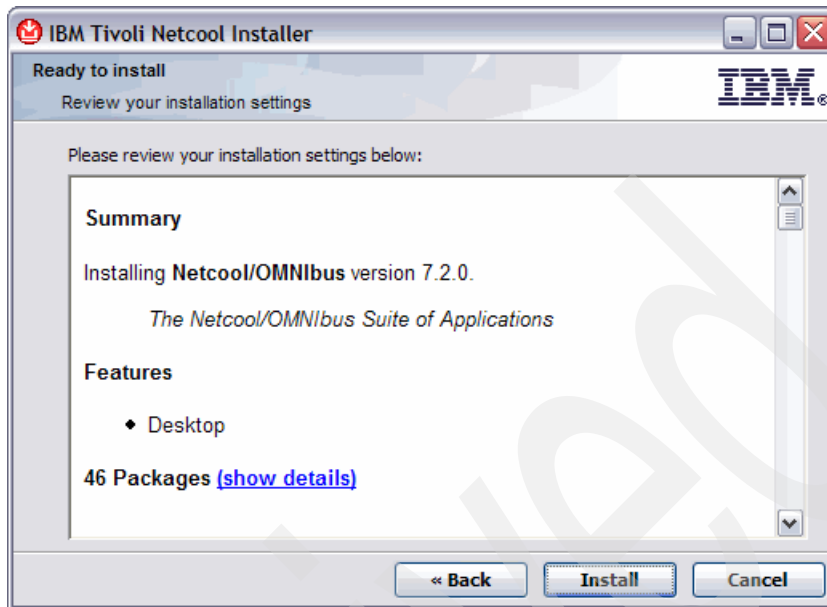


Figure 4-5 IBM Tivoli Netcool/OMNIBus installation

6. Reboot the machine at the end of the installation after clicking **Finish**, as shown in Figure 4-6.

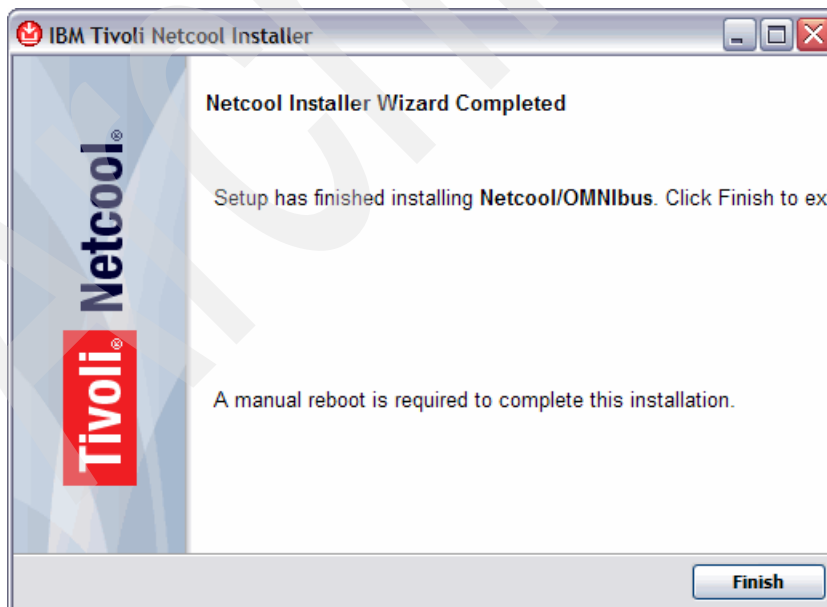


Figure 4-6 IBM Tivoli Netcool/OMNIBus installation complete

7. After the reboot, start the Server Editor by selecting **All Programs** → **Netcool Suite** → **Server Editor**. Create an entry for the ObjectServer in the Server Editor window on the remote desktop, as shown in Figure 4-7. If failover is enabled, then you must also specify the backup ObjectServer.

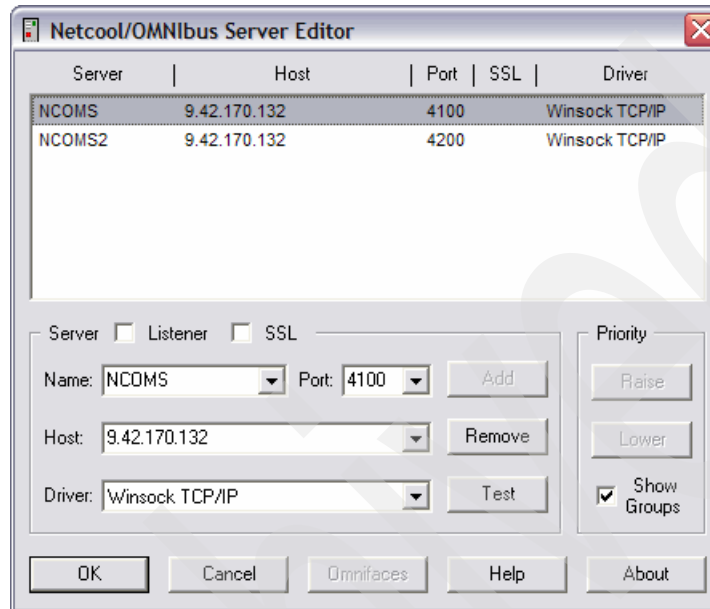


Figure 4-7 IBM Tivoli Netcool/OMNIBus Server Editor

8. Launch the event list by selecting **Start** → **All Programs** → **Netcool Suite** → **Event List** and log into the ObjectServer, as shown in Figure 4-8.

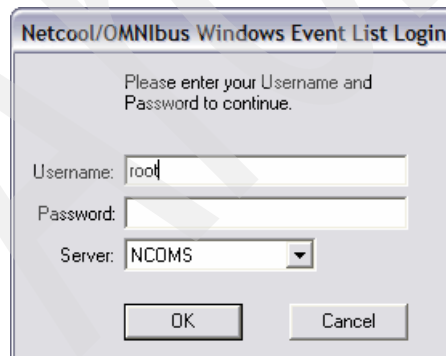


Figure 4-8 IBM Tivoli Netcool/OMNIBus Windows Event List login window

9. The desktop is shown in Figure 4-9 on page 91.

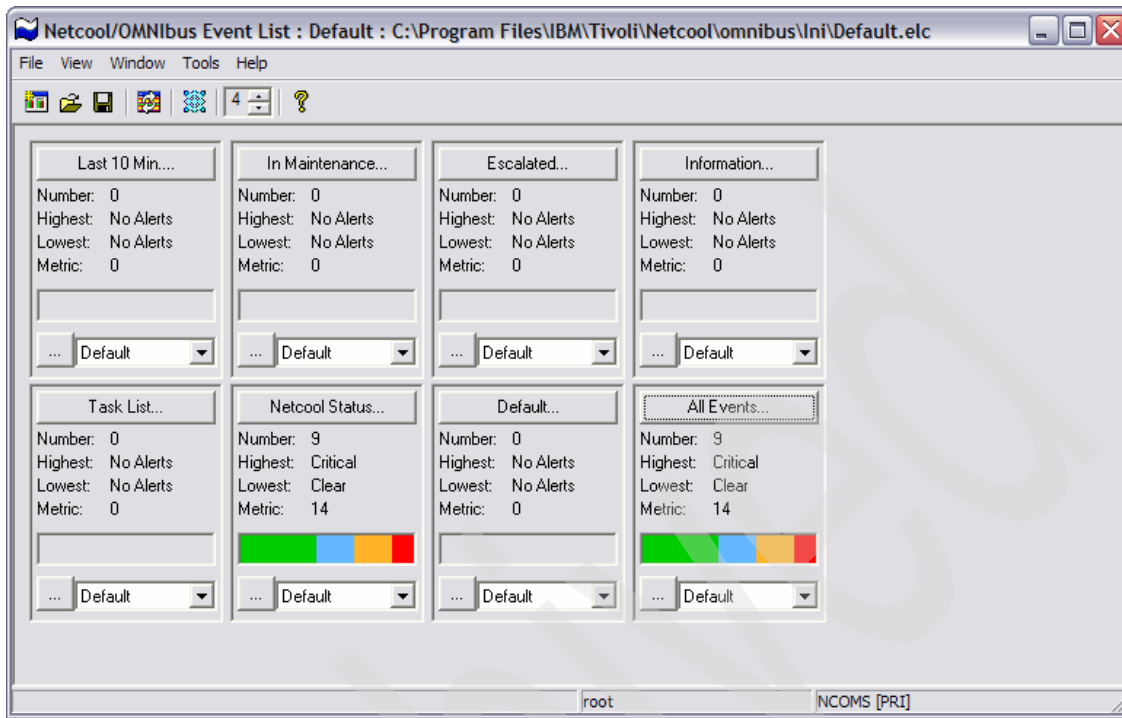


Figure 4-9 IBM Tivoli Netcool/OMNIBus Windows Even list default window

4.1.2 Gateway configuration

Gateways allows communication between IBM Tivoli Netcool/OMNIBus and external components, such as other ObjectServer or third-party applications. The gateway can operate in one of two modes: audit mode and reporter mode. You only need to run the gateway in reporter mode if you want the gateway to run with IBM Tivoli Netcool/Reporter. The mode in which the gateway runs is determined by the ODBC.ODBCGWType property in the properties file. Gateways also allow the copying of information besides alerts, such as user and group lists.

This section discusses the configuration of an uni-directional gateway. Perform the following steps:

1. Create a directory for the gateway files that indicates the gateway name. This example assumes that the gateway name would be UNI_GATE, so the directory is \$NCHOME/omnibus/gates/UNI_GATE.

2. Copy all files from the original gateway directory under \$NCHOME/omnibus/gates/. In our scenario, this would be objserv_uni or objserv_bi to \$NCHOME/omnibus/gates/UNI_GATE. The files for an uni-directional gateway are:
 - objserv_uni.props
 - objserv_uni.map
 - objserv_uni.reader.tblrep.def
 - objserv_uni.startup.cmd
3. Rename all the copied files so that they match the unique name of the gateway UNI_GATE.*. The file list then becomes:
 - UNI_GATE.props
 - UNI_GATE.map
 - UNI_GATE.reader.tblrep.def
 - UNI_GATE.startup.cmd
4. Edit the UNI_GATE.props file and modify the properties to what is shown in Example 4-1.

Example 4-1 UNI_GATE.props

```

Name           : 'UNI_GATE'
PropsFile      : '$OMNIHOME/gates/UNI_GATE/UNI_GATE.props'
Gate.MapFile   : '$OMNIHOME/gates/UNI_GATE/UNI_GATE.map'
Gate.StartupCmdFile : '$OMNIHOME/gates/UNI_GATE/UNI_GATE.startup.cmd'
Gate.Reader.Server : 'NCOMS'
Gate.Reader.TblReplicateDefFile :
'$OMNIHOME/gates/UNI_GATE/UNI_GATE.reader.tblrep.def'
Gate.Writer.Server : 'NCOMS2'
  
```

Some common properties are listed in Table 4-1.

Table 4-1 Common gateway properties

Property name	Description
Gate.Java.Arguments string	Use this property to specify the Java arguments.
Gate.Java.ClassPath string	Use this property to specify the path to the third-party and user-defined Java classes that create the required JVM™ environment.
Gate.Java.Debug string	Use this property to specify whether the debugging option is enabled for the JVM environment.
Gate.Java.LibraryPath string	Use this property to specify the path to the library files for the required JVM environment.

Property name	Description
Gate.Mapper.Debug string	Use this property to specify whether the gateway includes mapper debug messages in the debug log.
Gate.Mapper.Forward HistoricDetails string	Use this property to specify whether the gateway forwards all historic details after a converted update.
Gate.Mapper.Forward HistoricJournals string	Use this property to specify whether the gateway forwards all historic journals after a converted update.
Gate.Reader.Debug string	Use this property to specify whether the gateway includes gateway reader debug messages in the debug log.
Gate.Reader. Description string	Use this property to specify the application description for the reader connection. This description is used in triggers and allows you to determine which component of the gateway attempted to perform an action.
Gate.Reader.Details TableName string	Use this property to specify the name of the details table that the gateway reads. The default is alerts.details.
Gate.Reader.Failback Enabled string	Use this property to specify whether failback is enabled for the ObjectServer.
Gate.Reader.Failback Timeout integer	Use this property to specify the time (in seconds) that the gateway reader waits before entering failback mode.
Gate.Reader.Iduc FlushRate integer	Use this property to specify the rate (in seconds) of the granularity of the reader. If you set this property to 0, the reader gets its updates at the same granular rate as that of the ObjectServer to which it is connected.
Gate.Reader.Journal TableName string	Use this property to specify the name of the journal table that the gateway reads.
Gate.Reader.LogOSSql string	Use this property to specify whether the gateway logs all SQL commands sent to the ObjectServer in debug mode.
Gate.Reader.Password string	Use this property to specify the password associated with the user specified by the Gate.Reader.Username property.
Gate.Reader. ReconnectTimeout string	Use this property to specify the time (in seconds) between each reconnection poll attempt that the gateway makes if the connection to the ObjectServer is lost.
Gate.Reader.Server string	Use this property to specify the name of the ObjectServer from which the gateway reads alerts.
Gate.Reader. StatusTableName string	Use this property to specify the name of the status table that the gateway reads. The default is alerts.status.

Property name	Description
Gate.Reader. TblReplicateDefFile string	Use this property to specify the path to the table replication definition file.
Gate.Reader.Username string	Use this property to specify the user name that is used to authenticate the ObjectServer connection.
Gate.XMLGateway. Debug string	Use this property to specify whether the debug log includes the debug messages of the Message Bus Gateway.
Gate.XMLGateway. FailbackEnabled string	Use this property to specify whether failback is enabled for the Message Bus Gateway.
Gate.XMLGateway. FailbackTimeout string	Use this property to specify the time (in seconds) that the Message Bus Gateway awaits before entering failback mode.
Gate.XML Gateway.MessageID string	Use this property to specify the message ID for the Message Bus Gateway. The message ID determines which transformation is required; the message ID will either be a hardcoded value or @col_name, which then uses the value of the column specified as the ID to decide which transformation is required.
Gate.XMLGateway. ReconnectTimeout string	Use this property to specify the time (in seconds) between each reconnection poll attempt if the XML gateway loses the connection to the ObjectServer.
Gate.XMLGateway. TransformerFile string	Use this property to specify the path to the transformer file.
Gate.XMLGateway. TransportFile string	Use this property to specify the path to the transport file.
Gate.XMLGateway. TransportType	Use this property either to specify the transport method to be used (JMS or file) or to define the name of the transport module class to use.

Additional bi-directional gateway properties are listed in Table 4-2. These properties are prefixed by Gate.ObjectServerA or Gate.ObjectServerB. Each represents the source or target ObjectServer.

Table 4-2 Bi-directional gateway properties

Property name	Description
Buffersize integer	Use this property to specify the number of entries that the gateway stores in the buffer for this ObjectServer before flushing, if buffering is enabled. This property can be used to fine-tune the efficiency of the gateway. The default is 25.

Property name	Description
Debug Boolean	Use this property to specify whether the gateway includes debug messages for this ObjectServer in the gateway debug log. The default is TRUE.
DeletelfNoDedup Boolean	Use this property to specify whether a deletion is applied if the gateway forwards deletes. The default is FALSE.
Description string	Use this property to specify an application description for the connection to ObjectServer A. This description is used in triggers and allows you to determine which component of the gateway attempted to perform an action. The default is " ".
FailbackEnabled Boolean	Use this property to enable failback for this ObjectServer. The default is FALSE.
FailbackTimeout integer	Use this property to specify the time (in seconds) that the gateway allows before checking for the return of the master ObjectServer and failing back. The default is 30.
LogOSSql Boolean	Use this property to specify whether the gateway logs all SQL commands sent to this ObjectServer in debug mode. The default is FALSE.
Password string	Use this property to specify the password associated with the user specified by the Username property. The default is " ".
ReconnectTimeout integer	Use this property to specify the time (in seconds) between each reconnection poll attempt if the connection to this ObjectServer is lost. The default is 30.
RefreshCacheOnUpdate Boolean	Use this property to specify how the hash table cache is refreshed for this ObjectServer. The default is FALSE.
SAF Boolean	Use this property to specify whether the gateway stores all table entries if the destination ObjectServer is unavailable and to forward them when the ObjectServer becomes available again. The default is FALSE.
SAFFile string	Use this property to specify the name of the file that the gateway uses to store table entries while the destination ObjectServer is unavailable. The default is \$OMNIHOME/var/objserv_bi/ NCO_GATE_ObjectServerA.store.
Serverstring	Use this property to specify the name of this ObjectServer. The default is NCOMS.
Username string	Use this property to specify the user name that is used to authenticate connection to this ObjectServer. This user name is used to establish both the writer's IDUC connection and the subsidiary SQL command connection. The default is root.
TblReplicateDefFile string	Use this property to specify the path to the table replication definition file. The default is \$OMNIHOME/gates/objserv_bi/ objserv_bi.objectservera.tblrep.def.

Property name	Description
StatusTableName string	Use this property to specify the name of the status table that the gateway reads. The default is alerts.status.
JournalTableName string	Use this property to specify the name of the journal table that the gateway reads. The default is alerts.journal.
DetailsTableName string	Use this property to specify the name of the details table that the gateway reads. The default is alerts.details.
IDUCFlushRate integer	Use this property to specify the rate (in seconds) of the granularity of the reader. If you set this property to 0, the reader gets its updates at the same granular rate as that of the ObjectServer to which it is connected. The default is 0.

5. Define the actions that are used to specify a command that is run after an event has been processed by the gateway. This is specified using the statement AFTER IDUC D0.
6. Edit the interface definition file omni.dat and add the entry for the gateway, as shown in Example 4-2.

Example 4-2 Additions to the interface definitions file

```
[UNIGATE]
{
    Primary: 9.42.170.132 4300
}
```

7. Regenerate the interface file by running **nco_igen**.
8. Edit the UNI_GATE.map file and the UNI_GATE.reader.tbl rep.def file, and modify the entries to define which ObjectServer tables and fields are accessed by the gateway. Consider the sequence and modification of the alerts.status tables, as they must match exactly. A special field is defined for a desktop ObjectServer mapping, which is called MasterSerial, which refers to the master ObjectServer to which the desktop ObjectServer is pointing. A sample mapping of the alerts.status table is shown in Example 4-3 on page 97.

Example 4-3 Sample mapping

```
CREATE MAPPING StatusMap
(
  'Identifier'      = '@Identifier'      ON INSERT ONLY,
  'Node'           = '@Node'            ON INSERT ONLY,
  'NodeAlias'      = '@NodeAlias'      ON INSERT ONLY,
  ...
  'ServerName'     = '@ServerName'     ON INSERT ONLY,
  'ServerSerial'   = '@ServerSerial'   ON INSERT ONLY,
  'MasterSerial'   = '@Serial'         ON INSERT ONLY
);
```

9. Start a ObjectServer gateway. The executable for the uni-directional gateway is `nco_g_objserv_uni`, while the executable for the bi-directional gateway is `nco_g_objserv_bi`. The gateway executable accept these parameters:

-name For the gateway name (UNI_GATE in our example)
-propsfile The property file UNI_GATE.props in
 \$NCHOME/omnibus/gates/UNI_GATE/.

10. All gateway and client connections can be queried from the ObjectServer database. Using `nco_sql`, all connections are can be queried from the `catalog.connections` table.

11. Verify that the gateway is working by deleting and or modifying an event on the one ObjectServer and checking to see that the modification is made on the other ObjectServer:
 - a. Open two separate event lists on both servers, as shown in Figure 4-10.

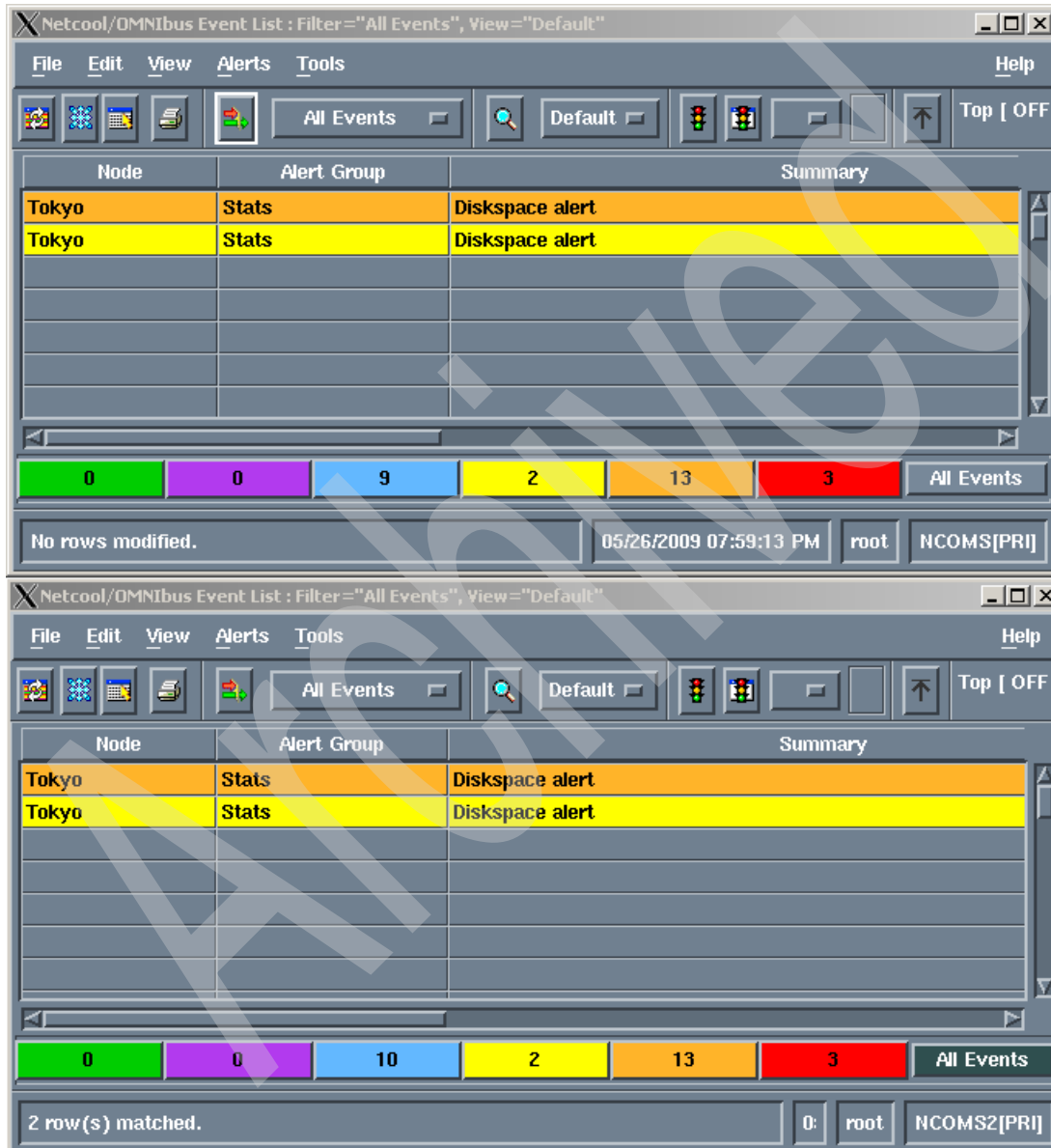


Figure 4-10 AEL on both servers

b. Delete an event on the first server, as shown in Figure 4-11.

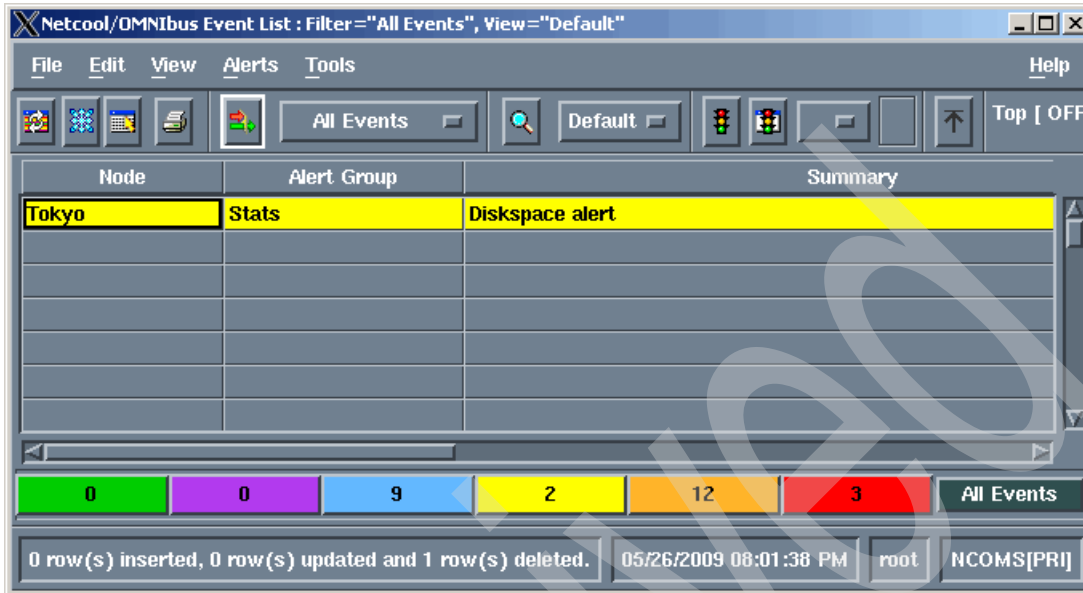


Figure 4-11 AEL on NCOMS

- c. Check that the modification is passed to the other ObjectServer, as shown in Figure 4-12

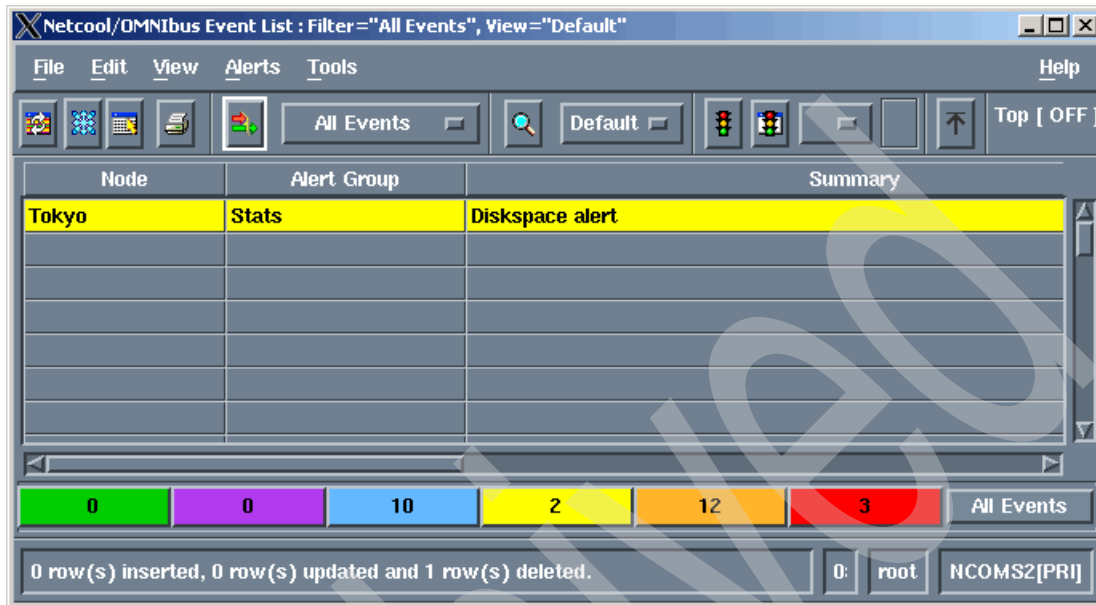


Figure 4-12 AEL on NCOMS2

4.1.3 Process agent configuration

If you use the process agent (sometimes called process control), operating IBM Tivoli Netcool/OMNIBus components becomes easier, as the startup and shutdown of the component is controlled by the process agent. This section discusses the configuration of the process agent:

1. Open the `$NCHOME/omnibus/etc/nco_pa.conf` file under the `nco_process` stanza, and verify and match the name of the ObjectServer definition, as shown in Example 4-4 on page 101.

Example 4-4 nco_pa.conf, nco_process

```
nco_process 'MasterObjectServer'
{
  Command '$OMNIHOME/bin/nco_objserv -name NCOMS -pa NCO_PA' run as 0
  Host      = 'omnibus'
  Managed   = True
  RestartMsg = '${NAME} running as ${EUID} has been restored on
${HOST}.'
  AlertMsg  = '${NAME} running as ${EUID} has died on ${HOST}.'
  RetryCount = 0
  ProcessType = PaPA_AWARE
}
```

2. Under the `nco_routing` stanza that defines the communication and authentication to a process agent (remote and local) (shown in Example 4-5):
 - Set the `Host` value to the host name of the local machine.
 - Set the user and password values under `nco_routing` if using secure mode; the user must belong to the `ncoadmin` group.

Note: The password should then be encrypted using the `nco_pa_crypt` command.

Example 4-5 nco_pa.conf, nco_routing

```
nco_routing
{
  host 'omnibus' 'NCO_PA' 'user' 'password'
}
```

3. The `nco_service` definition specifies the services that the process agent must manage. A sample `nco_service` definition is shown in Example 4-6.

Example 4-6 Sample nco_service

```
nco_service 'Omnibus'
{
  ServiceType = Master
  ServiceStart = Non-Auto
  process 'ObjectServer' NONE
  process 'Proxy' 'ObjectServer'
  process 'Probe' 'Proxy'
  process 'Probe-1' 'ObjectServer'
  process 'Sleep' 5
}
```

Where:

nco_service	Defines the name of the service. Each service name must be unique within the process control network.
ServiceType	Defines whether this service should be started before all other services and handled as the master service upon which other services depend. This can be set as either Master or Non-Master .
ServiceStart	This can be set to Auto to start the service as soon as <code>nco_pad</code> has started, and Non-Auto if the service must be started manually with the <code>nco_pa_start</code> command.
process	Each process entry defines a process that must be run as part of the service. The second and subsequent entries represents the process dependencies, so a process cannot start unless another is already running. You would typically start the <code>ObjectServers</code> , then gateways and then the probes.

4. Add an entry to the interface file `$NCHOME/etc/omni.dat` for `nco_pa` and run `nco_igen`. The entry is shown in Example 4-7.

Example 4-7 omni.dat gateway configuration

```
[NCO_PA]
{
  Primary: 9.42.170.132 4400
}
```

5. Start the process agent by running `nco_pad` under `$NCHOME/omnibus/bin/`. The resulting messages are shown in Example 4-8 on page 103.

Example 4-8 nco_pad

Netcool/OMNIBus Process Agent Daemon - Version 7.2

Netcool/OMNIBus PA API Library Version 7.2
Sybase Server-Library Release: 15.0

Server Settings :

Name of server : NCO_PA
Path of used log file :
/opt/netcool//omnibus/log/NCO_PA.log
Configuration File :
/opt/netcool//omnibus/etc/nco_pa.conf
Child Output File : /dev/null
Maximum logfile size : 1024
Thread stack size : 69632
Message Pool size : 45568
PID Message Pool size : 50
Rogue Process Timeout : 30
Truncate Log : False
Instantiate server to daemon : True
Internal API Checking : False
No Configuration File : False
Start Auto-start services : True
Authentication System : UNIX
Trace Net library : False
Trace message queues : False
Trace event queues : False
Trace TDS packets : False
Trace mutex locks : False
Host DNS name : omnibus
PID file (from \$OMNIHOME) : ./var/nco_pa.pid
Kill Process group : False
Secure Mode : False
Administration Group Name. : ncoadmin

Forking to a Daemon Process.....

6. Verify the status of the process agent. Run **nco_pa_status** under `$NCHOME/omnibus/bin/`; you can qualify the status by selecting the ObjectServer name by using the `-server` parameter. You should also identify yourself with the authority to access the server by using the `-user` parameter. The user specified must be a member of `ncoadmin` and can be authenticated on the operating system level by process agent (which is usually run by root).

7. Manually kill one of the running ObjectServer processes using the **kill** command.
8. Run **nco_pa_status** again to verify that the killed process is running with a different process ID, as shown in Example 4-9. This verifies that the process agent restarts the killed process.

Example 4-9 nco_pa_status

```
[netcool@omnibus etc]$ nco_pa_status -user netcool
Login Password:
-----
Service Name  Process Name                Hostname  User      Status  PID
-----
Core          MasterObjectServer          omnibus  netcool   RUNNING 29487
              SecondaryObjectServer       omnibus  netcool   RUNNING 29488
              BackupObjectServer          omnibus  netcool   RUNNING 29491
-----
[netcool@omnibus etc]$ kill -TERM 29487
[netcool@omnibus etc]$ nco_pa_status -user netcool
Login Password:
-----
Service Name  Process Name                Hostname  User      Status  PID
-----
Core          MasterObjectServer          omnibus  netcool   RUNNING 29690
              SecondaryObjectServer       omnibus  netcool   RUNNING 29488
              BackupObjectServer          omnibus  netcool   RUNNING 29491
-----
```

Errors in the process agent can be found in the log file in `$NCHOME/omnibus/log/<pa_name>.log`.

4.1.4 Startup script

Once the customization of the components are done, we can define the startup script that allows the processes to be started automatically when the server starts. Perform the following steps:

1. Execute the start-up script `$NCHOME/omnibus/install/startup/<arch>install`, which allows the definition for the process agent to start automatically when the system starts. This is described in Example 4-10 on page 105. Be sure to:
 - Input/verify the process agent name.
 - Select secure mode or not.
 - Enter a value for the `netcool_license_file` variable.

Example 4-10 Startup script installation

```
[root@omnibus startup]# cd /opt/netcool/omnibus/install/startup
[root@omnibus startup]# ./linux2x86install
This script copies a startup script into the /etc/init.d directory to
enable you to automatically start and stop Netcool/OMNIBUS processes.
It does this by:
    Copying linux2x86/etc/rc.d/init.d/nco to /etc/init.d/nco
    Running "/sbin/chkconfig --add nco"

Do you wish to continue (y/n)? [y] y
Name of the Process Agent Daemon [NCO_PA]:
Should NCO_PA run in secure mode (y/n)? [y] n
Enter value for environment variable NETCOOL_LICENSE_FILE if required
[27000@localhost]:
Scripts installed.
```

2. Modify some of the environment variables for the startup script nco, as shown in Example 4-11.

Example 4-11 The nco startup script variables

```
# Variables that can be changed
START_NCO=Y # Start nco_pad
NCHOME=/opt/netcool/ # Set directory for $NCHOME
OMNIHOME=/opt/netcool/omnibus # Set directory for $OMNIHOME
NCO_PA="NCO_PA" # Set Process Agent's name
SECURE=n # Y/N run Process Agent in secure mode
NETCOOL_LICENSE_FILE=27000@localhost # Points to flex license server

export NCHOME OMNIHOME NCO_PA NETCOOL_LICENSE_FILE
```

3. The startup script nco must be linked to the different run levels on the system, as shown in Example 4-12.

Example 4-12 Startup symbolic links

```
[root@omnibus etc]# cd /etc/
[root@omnibus etc]# ls -l rc?..d/*nco*
lrwxrwxrwx 1 root root 13 May 28 21:59 rc0.d/K65nco -> ../init.d/nco
lrwxrwxrwx 1 root root 13 May 28 21:59 rc1.d/K65nco -> ../init.d/nco
lrwxrwxrwx 1 root root 13 May 28 21:59 rc2.d/K65nco -> ../init.d/nco
lrwxrwxrwx 1 root root 13 May 28 21:59 rc3.d/S81nco -> ../init.d/nco
lrwxrwxrwx 1 root root 13 May 28 21:59 rc4.d/K65nco -> ../init.d/nco
lrwxrwxrwx 1 root root 13 May 28 21:59 rc5.d/S81nco -> ../init.d/nco
lrwxrwxrwx 1 root root 13 May 28 21:59 rc6.d/K65nco -> ../init.d/nco
```

4. Verify the operation of the startup script by running the nco script, as shown in Example 4-13. Be sure to:
 - Run the command `/etc/init.d/nco start` and verify that the process agent has started.
 - Run `/etc/init.d/nco stop` and verify that process agent (nco_pa) has stopped.

Example 4-13 nco start and nco_pa_status command

```
[root@omnibus etc]# /etc/init.d/nco start
Netcool/OMNIBus : Starting Process Control ... [ OK ]
[root@omnibus etc]# /opt/netcool/omnibus/bin/nco_pa_status -server NCO_PA
Login Password:
```

Service Name	Process Name	Hostname	User	Status	PID
Core	MasterObjectServer	omnibus	netcool	RUNNING	30044
	SecondaryObjectServer	omnibus	netcool	RUNNING	30046
	BackupObjectServer	omnibus	netcool	RUNNING	30047

```
[root@omnibus etc]# /etc/init.d/nco stop
Netcool/OMNIBus : Stopping Process Control ... [ OK ]
[root@omnibus etc]# /opt/netcool/omnibus/bin/nco_pa_status -server NCO_PA
Login Password:
May 28 22:02:17 2009: Error: Failed to make a connection to NCO_PA.
```

When all IBM Tivoli Netcool/OMNIBus processes are run under the process agent, you must start and stop them using the `nco_pa_start` and `nco_pa_stop` commands; otherwise, the process agent would always attempt to maintain the state of execution of the process. Non-OMNIBus processes that are defined under process agent may fail, as they are not aware of the process agent control or some environment variables may be missing.

4.2 Security configuration

Security configuration in IBM Tivoli Netcool/OMNIBus concerns defining users, groups and roles. The discussion includes:

- ▶ 4.2.1, “Role creation” on page 107
- ▶ 4.2.2, “Group creation” on page 112
- ▶ 4.2.3, “User creation” on page 115

Note: User, group, and role names can be any text string up to 64 characters in length and can include spaces.

4.2.1 Role creation

Roles define the authority that a user or a group is allowed in order to access a facility in IBM Tivoli Netcool/OMNIBus. To create a role, perform the following steps:

1. Within the Configuration Manager, click the **Users** tab, as shown in Figure 4-13.

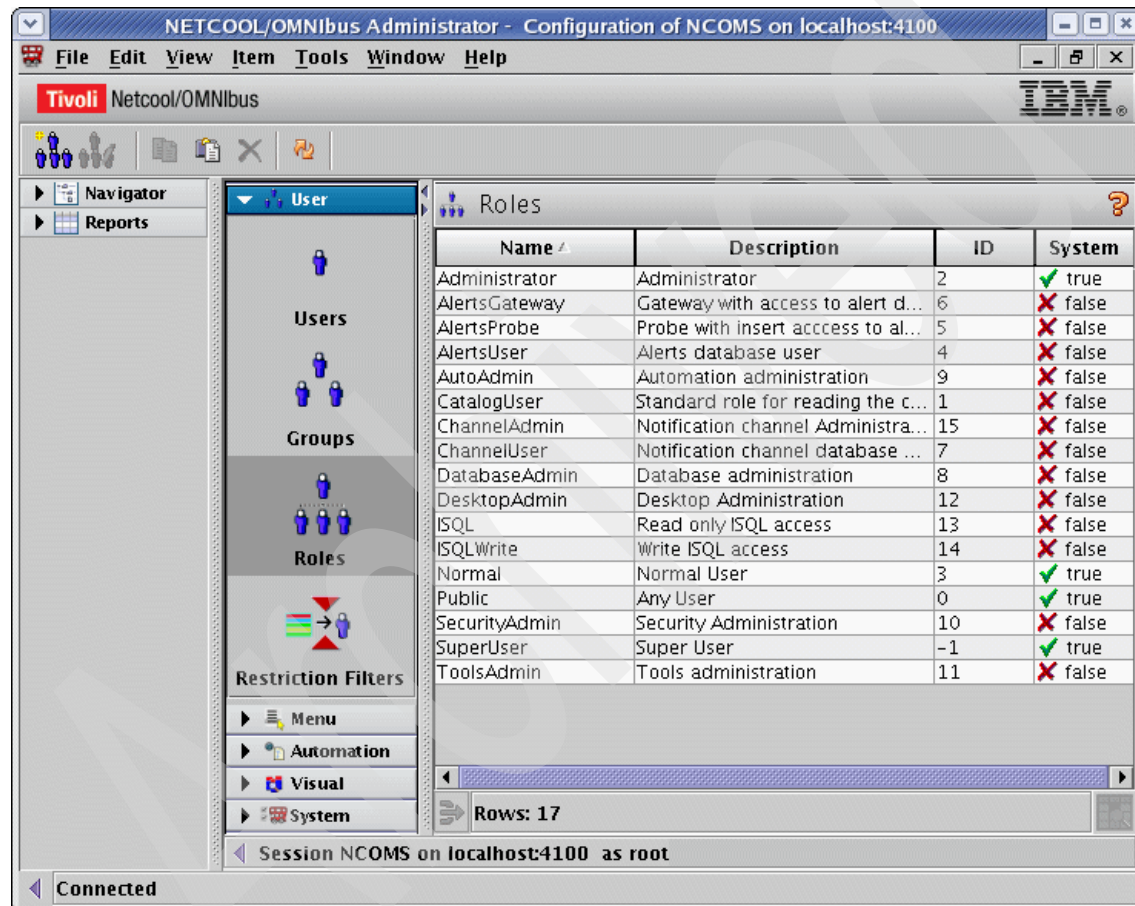



Figure 4-13 IBM Tivoli Netcool/OMNIBus Administrator Roles tab

- To create a role, click the Add New Role icon (). The New Role tab is shown in Figure 4-14.

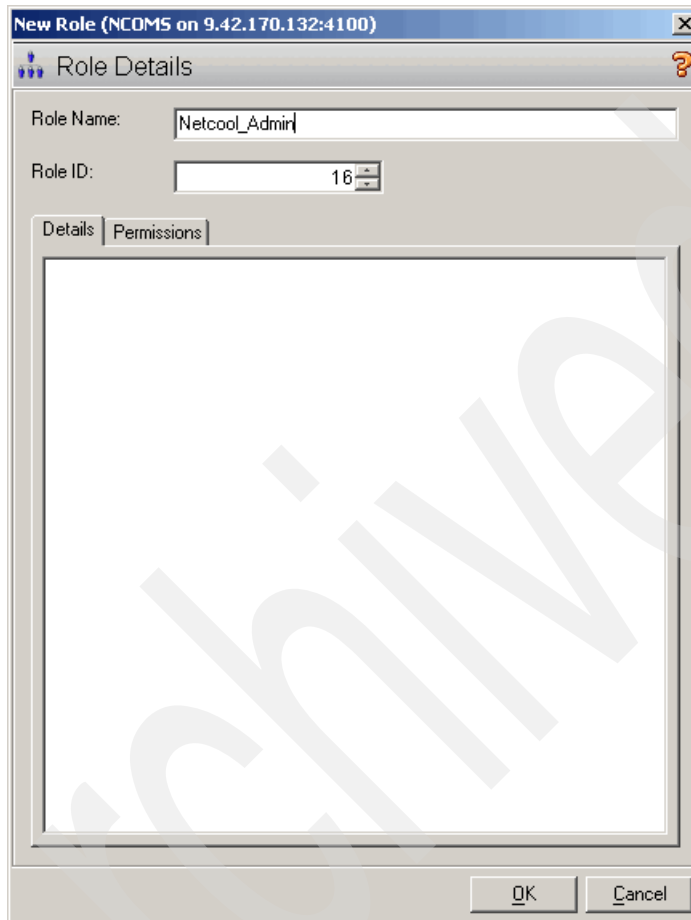


Figure 4-14 IBM Tivoli Netcool/OMNibus Administrator New Role tab

3. Assign role permissions, as shown in Figure 4-15.

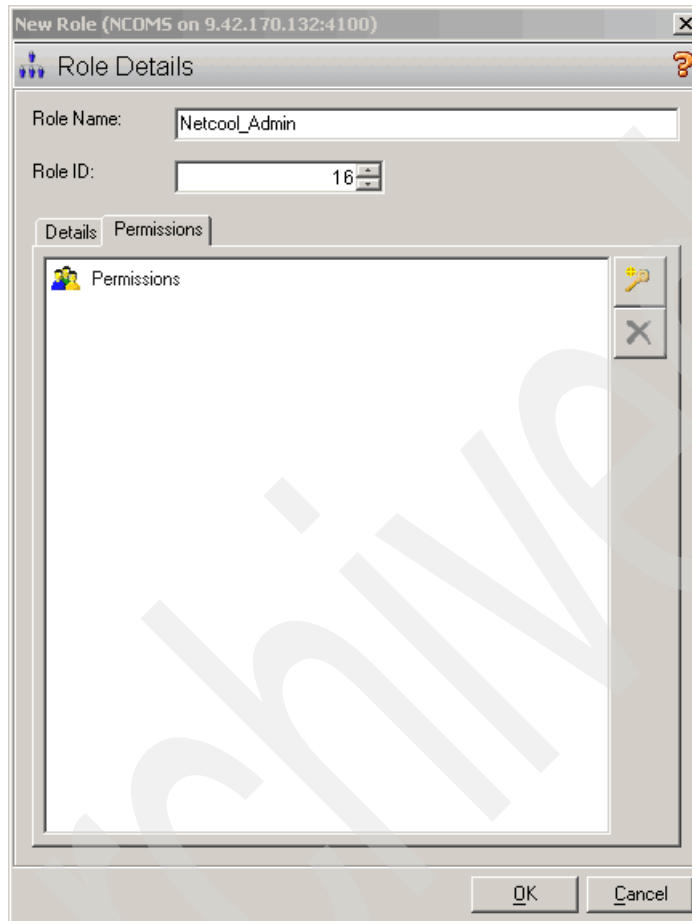


Figure 4-15 Permission tab

4. Click the new authority icon (🔑) and select an existing object, as shown in Figure 4-16.

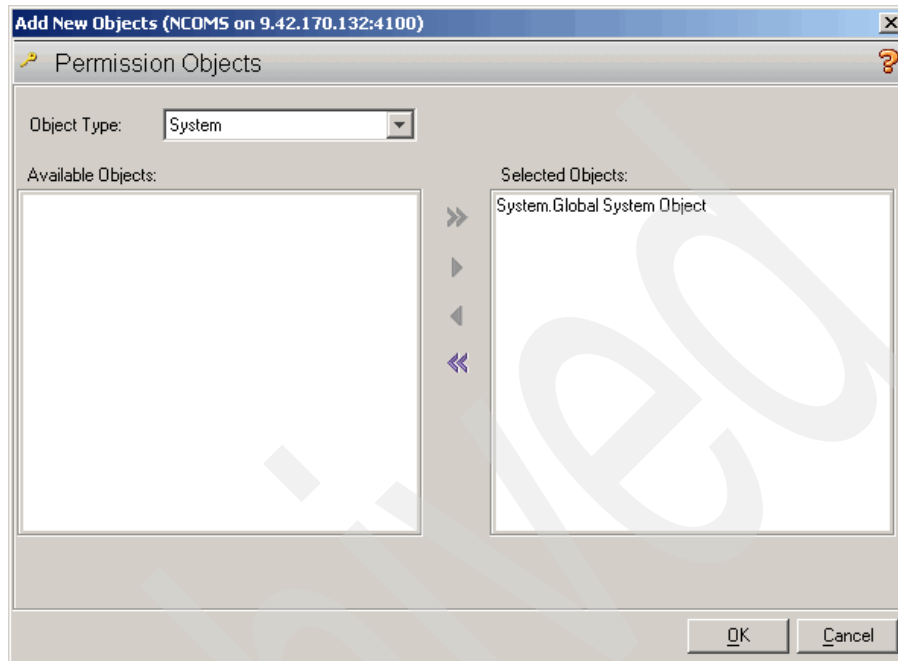


Figure 4-16 Permission Objects

5. The new permission object is shown in Figure 4-17 on page 111. You can then set the appropriate authorization. The authorization are similar to SQL authorization, such as SELECT, INSERT, DELETE, DROP, ALTER, and so on.

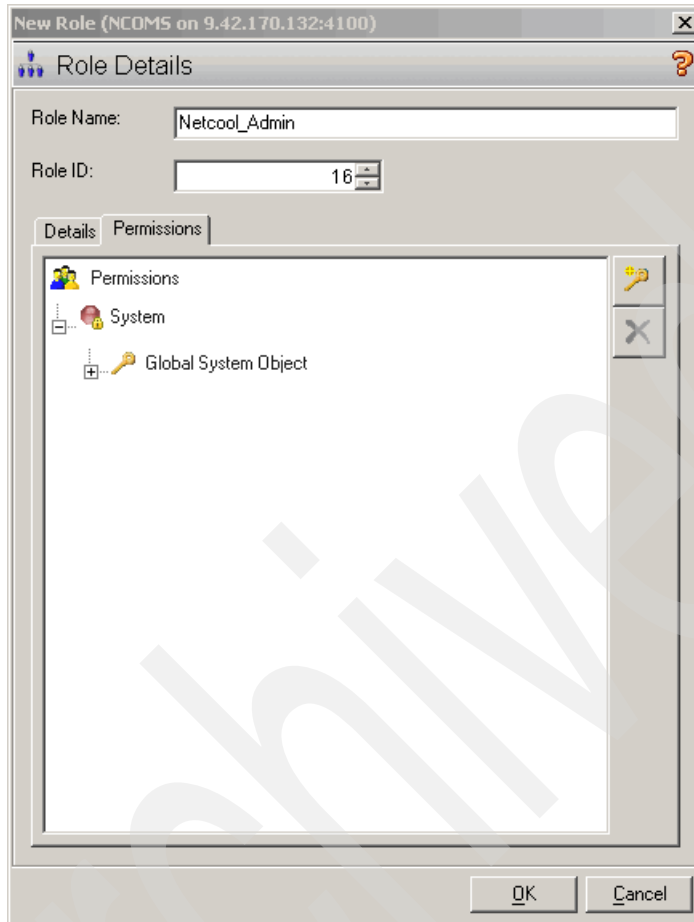


Figure 4-17 Permissions tab

6. Click **OK** and your new role is completed.

4.2.2 Group creation

Groups can also be created to provide role mapping for the individual user. In this case, users do not need to be assigned an individual role, because it would make administration difficult. To create a group in IBM Tivoli Netcool/OMNIBus, perform the following steps:

1. From the main IBM Tivoli Netcool/OMNIBus Administrator window, click the **Users** tab and then click **Groups**, as shown in Figure 4-18.

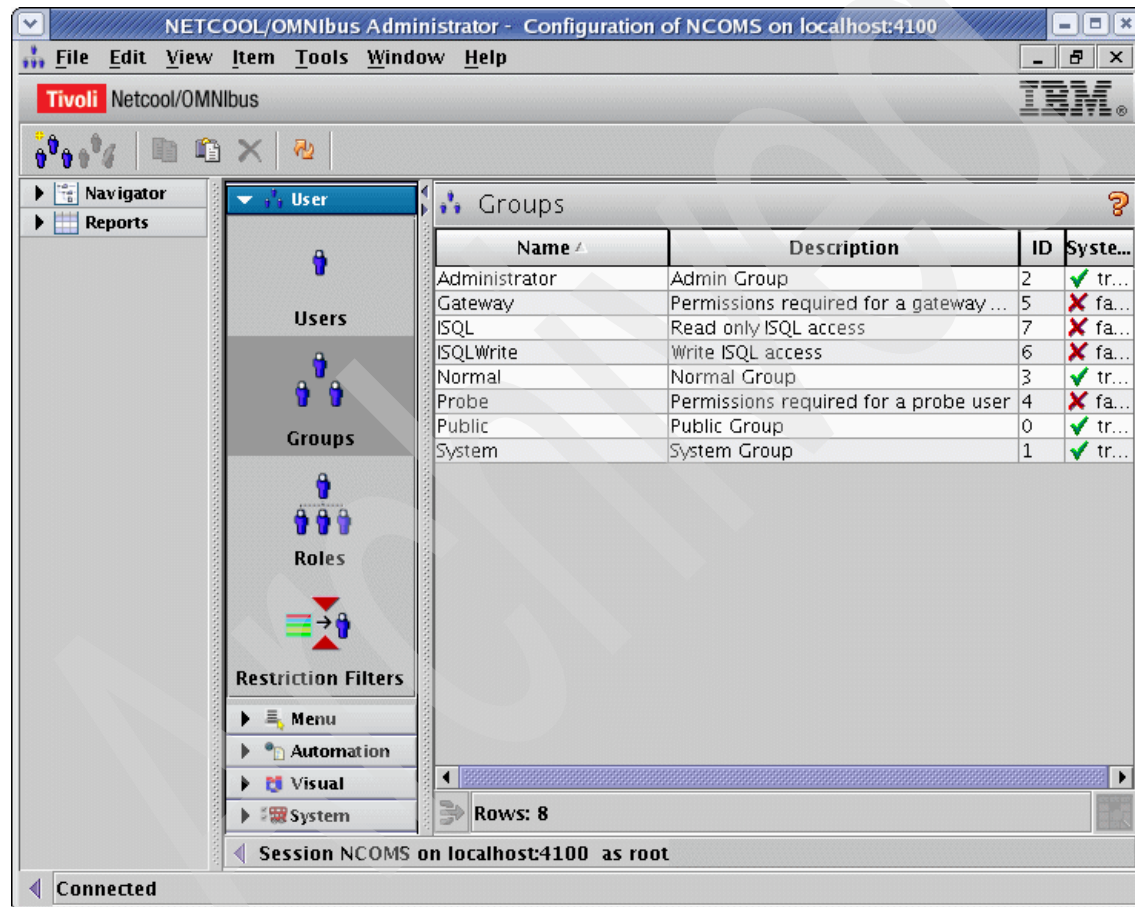



Figure 4-18 IBM Tivoli Netcool/OMNIBus Administrator Groups tab

2. Click the Add Group icon (). The Group Details window opens, as shown in Figure 4-19. Be sure to:
 - Assign a name to the group.
 - Assign an unused Group ID.
 - Assign a Role.

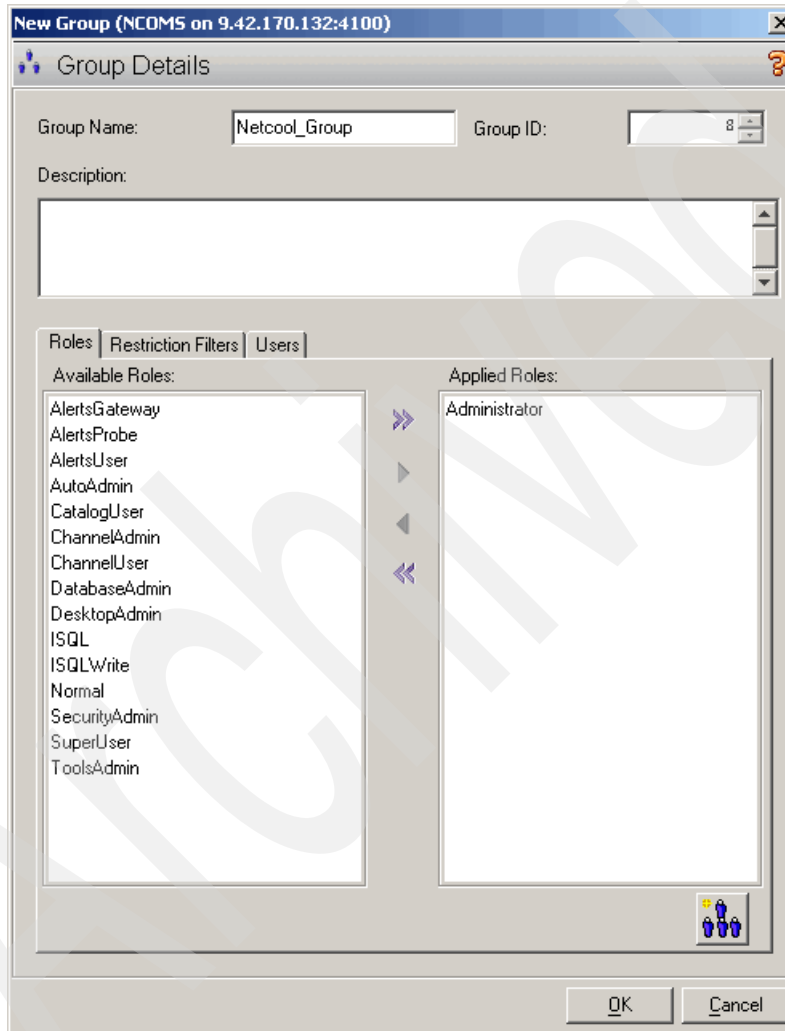


Figure 4-19 IBM Tivoli Netcool/OMNibus Administrator adding groups

3. You can assign a Restriction Filter, as shown in Figure 4-20.

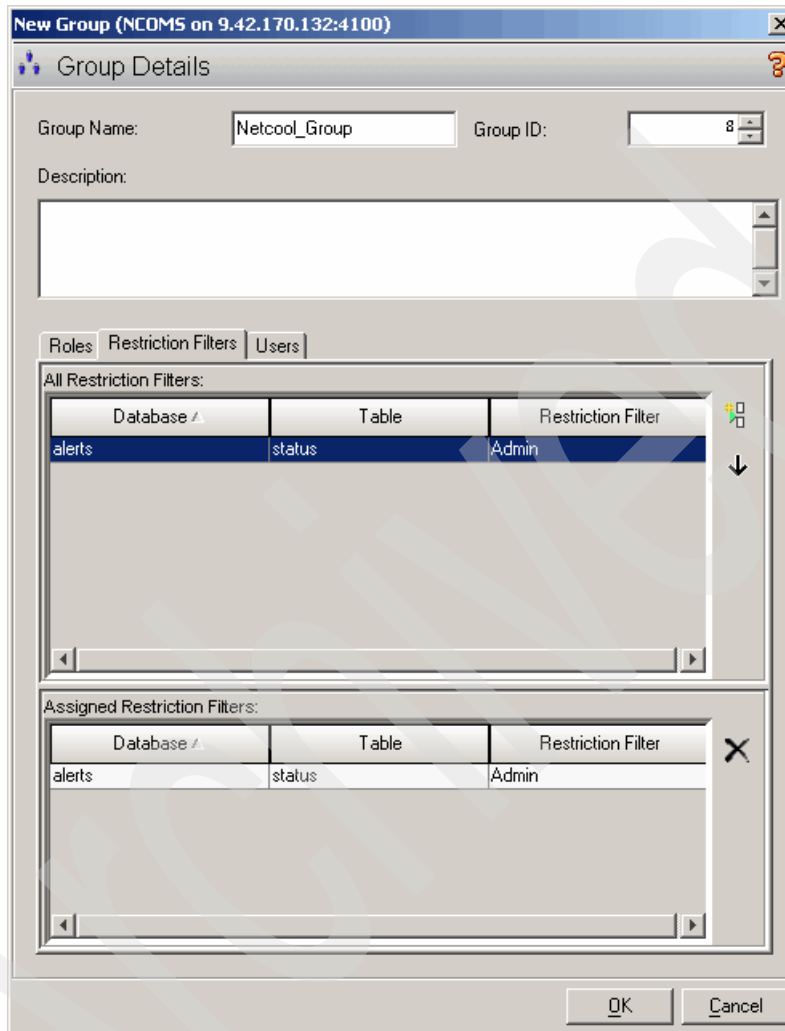


Figure 4-20 IBM Tivoli Netcool/OMNibus Administrator Restriction Filter tab

4. Click **OK**. The group is now created.

4.2.3 User creation

User creation is defined within IBM Tivoli Netcool/OMNIBus. The procedure for creating a user is as follows:

1. In the main IBM Tivoli Netcool/OMNIBus Administrator window, select **User** → **Users**, as shown in Figure 4-21.

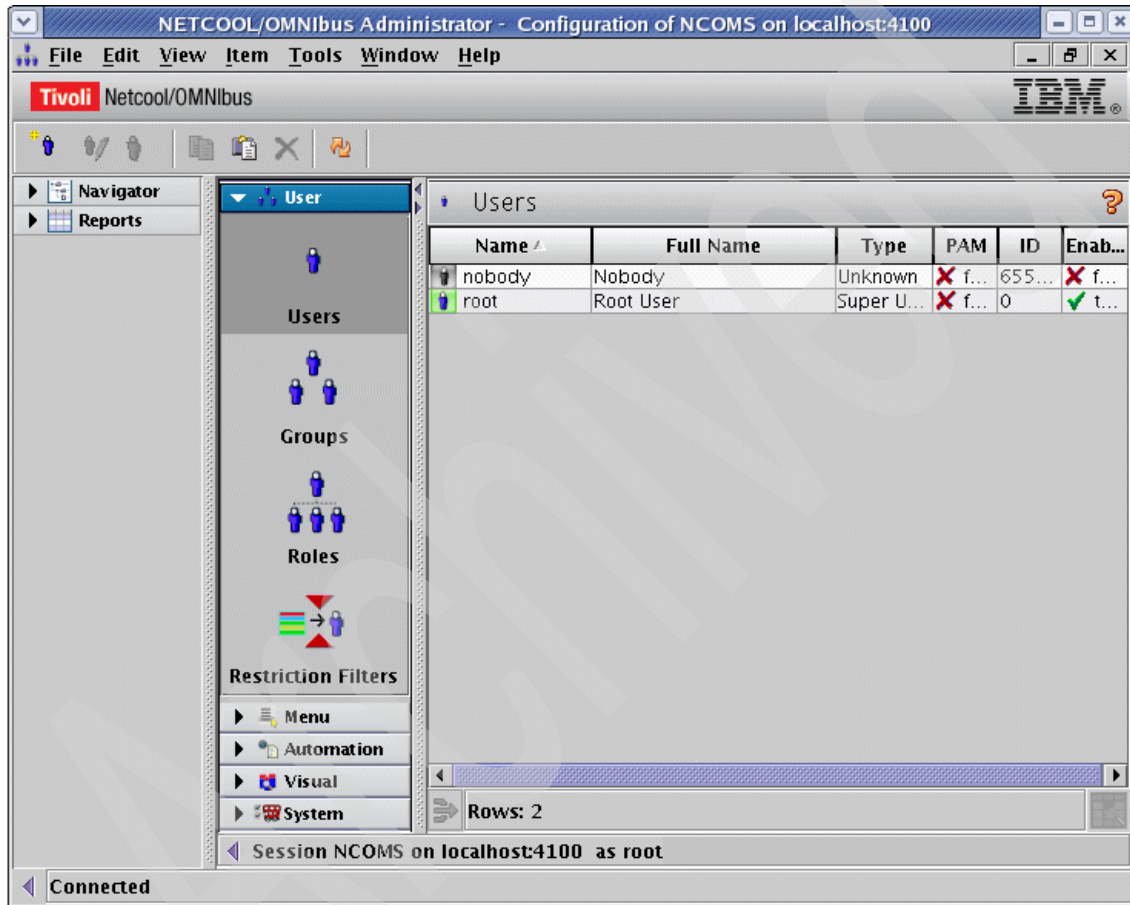



Figure 4-21 IBM Tivoli Netcool/OMNIBus Administrator windows Users tab

2. Click the Add User icon () in the toolbar. The Add User window opens, as shown in Figure 4-22. Be sure to:
 - Enter a Username and select an unused User ID.
 - Enter a FullName.
 - Check the **Create Conversion** check box

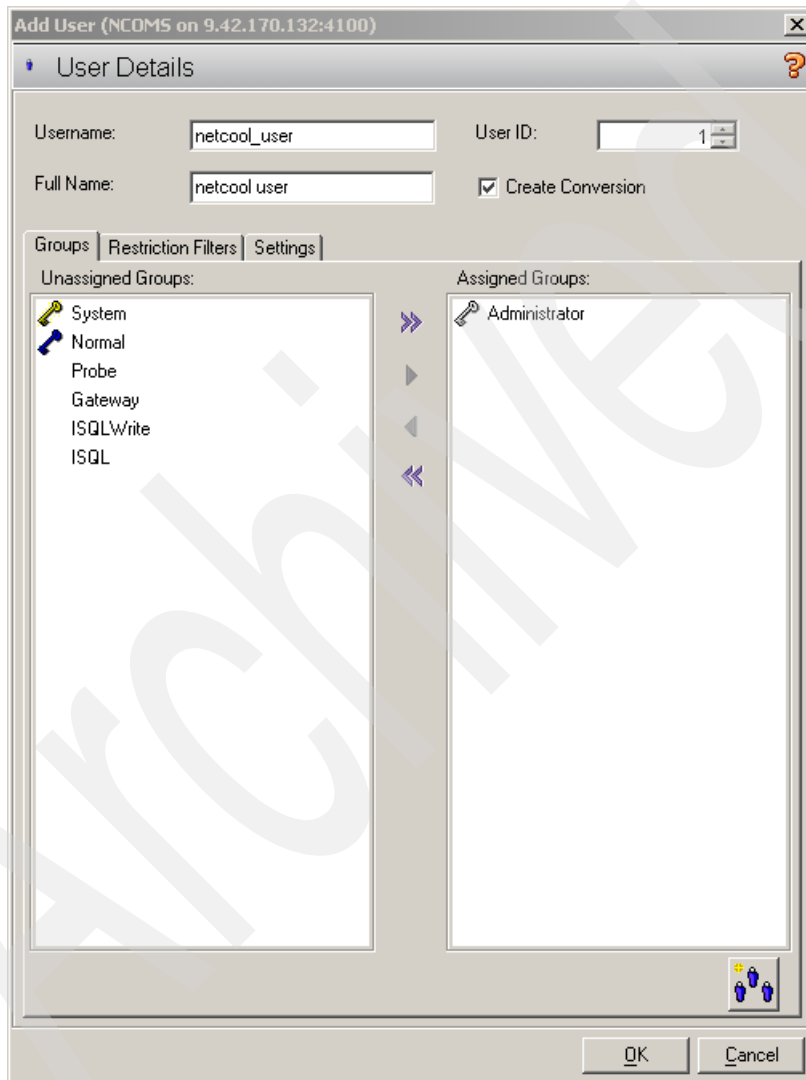


Figure 4-22 IBM Tivoli Netcool/OMNIBus Administrator Add User

3. In the Groups tab, double-click **Administrator** to assign the administrator role. Click **OK**.

4.3 ObjectServer customization


Once the ObjectServer is created and running, there are several customizations that can be performed. They are:

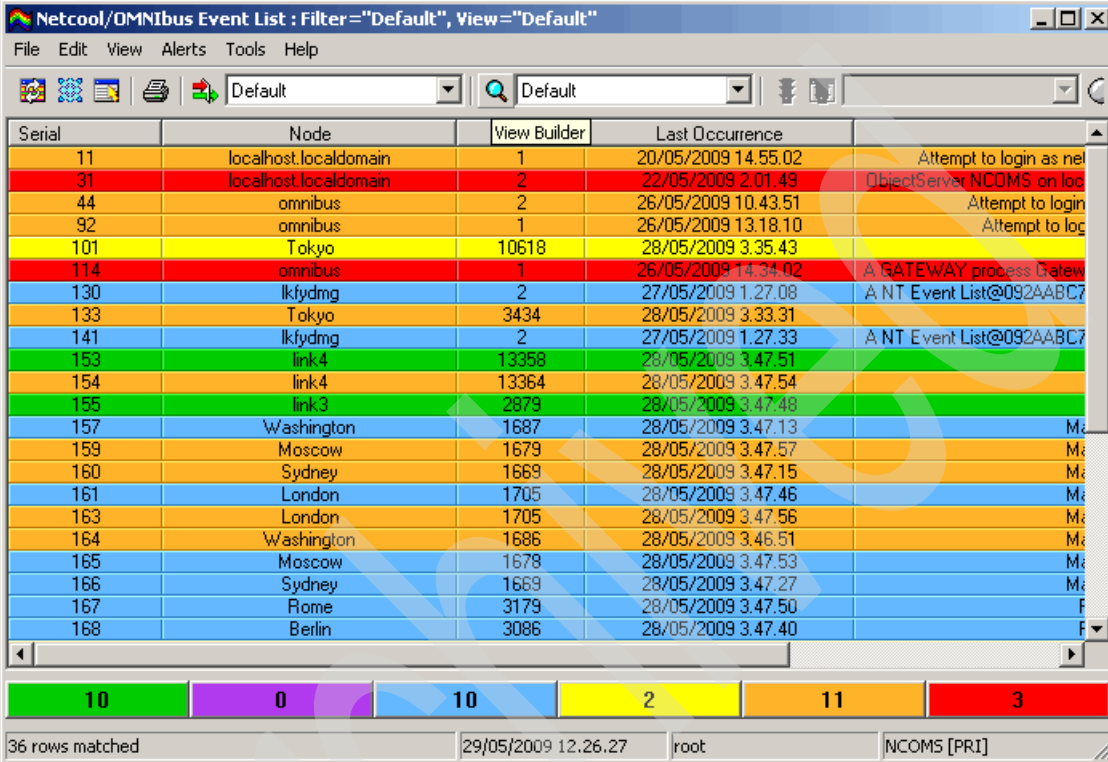
- ▶ 4.3.1, “View creation” on page 117
- ▶ 4.3.2, “Filters definition” on page 120
- ▶ 4.3.3, “Menu creation” on page 124
- ▶ 4.3.4, “Database field definition” on page 128
- ▶ 4.3.5, “Tool definition” on page 133
- ▶ 4.3.6, “Trigger creation” on page 140
- ▶ 4.3.7, “Class creation” on page 143

4.3.1 View creation

Depending on your requirements, an additional event view may be needed. This section describes view creation.

Perform the following steps:

1. From the Event list shown in Figure 4-23, launch View Builder by clicking the  button.



Serial	Node	View Builder	Last Occurrence	
11	localhost.localdomain	1	20/05/2009 14.55.02	Attempt to login as net
31	localhost.localdomain	2	22/05/2009 2.01.49	ObjectServer NCOMS on loc
44	omnibus	2	26/05/2009 10.43.51	Attempt to login
92	omnibus	1	26/05/2009 13.18.10	Attempt to log
101	Tokyo	10618	28/05/2009 3.35.43	
114	omnibus	1	28/05/2009 14.34.02	A GATEWAY process Gatew
130	lkfydmg	2	27/05/2009 1.27.08	A NT Event List@092AABC7
133	Tokyo	3434	28/05/2009 3.33.31	
141	lkfydmg	2	27/05/2009 1.27.33	A NT Event List@092AABC7
153	link4	13358	28/05/2009 3.47.51	
154	link4	13364	28/05/2009 3.47.54	
155	link3	2879	28/05/2009 3.47.48	
157	Washington	1687	28/05/2009 3.47.13	M
159	Moscow	1679	28/05/2009 3.47.57	M
160	Sydney	1669	28/05/2009 3.47.15	M
161	London	1705	28/05/2009 3.47.46	M
163	London	1705	28/05/2009 3.47.56	M
164	Washington	1686	28/05/2009 3.46.51	M
165	Moscow	1678	28/05/2009 3.47.53	M
166	Sydney	1669	28/05/2009 3.47.27	M
167	Rome	3179	28/05/2009 3.47.50	F
168	Berlin	3086	28/05/2009 3.47.40	F

36 rows matched | 29/05/2009 12.26.27 | root | NCOMS [PRI]

Figure 4-23 Event List window

2. In the View Builder window (Figure 4-24 on page 119), select **File** → **New** and:
 - Enter a view name.
 - In the Display Columns section, select the fields from Available Field. We choose Node, Severity, and Summary.
 - In the Sort Columns section, select the fields from Available Sort Fields pane and choose the order. We choose to sort by Severity in descending order.
 - The Restrict Row function limits the number of displayed rows.
 - The Set from Event List check box, when checked, would prevent the Restrict Row function from being overridden.

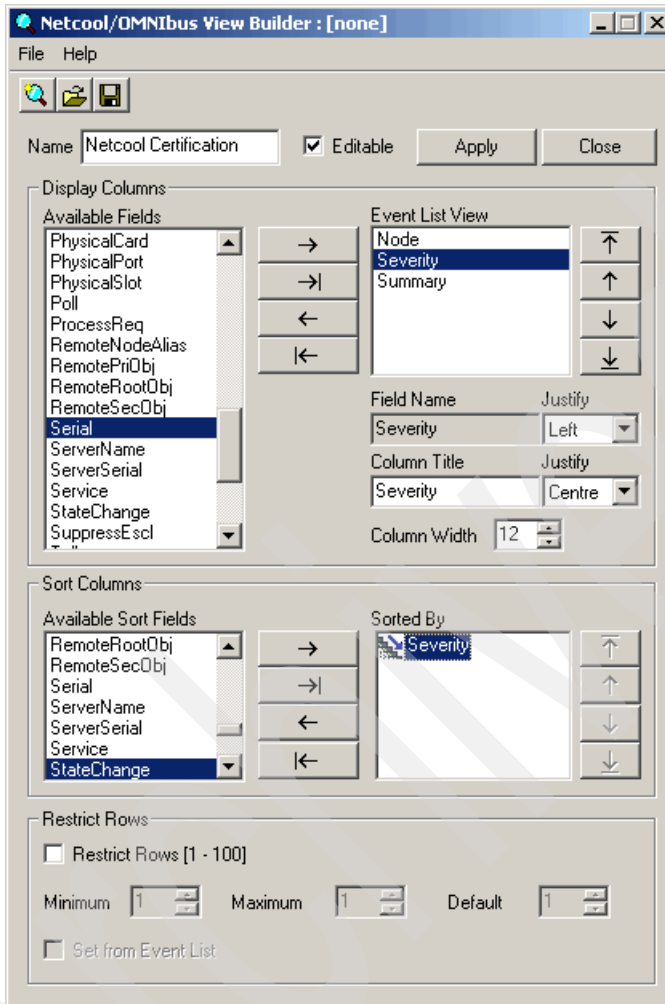


Figure 4-24 View Builder view

- Click **Apply** and click **Close**. You can now see the new view, as shown in Figure 4-25.

Node	Severity	Summary
omnibus	Critical	A GATEWAY process Gateway Reader/Writer running on omnibus has disco...
localhost.localdomain	Critical	ObjectServer NCOMS on localhost.localdomain shutdown at Fri May 22 08:0...
omnibus	Critical	A GATEWAY process Gateway Reader running on omnibus has disconnecte...
London	Major	Machine has gone offline
omnibus	Major	Attempt to login as root from host omnibus failed
Beijing	Major	Diskspace alert
Washington	Major	Machine has gone offline
Moscow	Major	Machine has gone offline
Tokyo	Major	Diskspace alert
omnibus	Major	Attempt to login as netcool from host omnibus failed
Sydney	Major	Machine has gone offline
link1	Major	Link Down on port
localhost.localdomain	Major	Attempt to login as netcool from host localhost.localdomain failed
link6	Major	Link Down on port
link5	Major	Link Down on port
link4	Major	Link Down on port
Beijing	Minor	Diskspace alert
Tokyo	Minor	Diskspace alert
London	Warning	Machine has gone online
Sydney	Warning	Machine has gone online
Berlin	Warning	Port failure : port reset
lkfydmg	Warning	A NT Event List@092AABC7 process running on lkfydmg has connected as...
IBM-5E6948F4416	Warning	A NT Event List@092CA843 process running on IBM-5E6948F4416 has co...

Summary: 8 Critical, 0 Major, 10 Warning, 2 Minor, 13 Informational, 3 Error

36 rows matched | 29/05/2009 12.31.56 | root | NCOMS [PRI]

Figure 4-25 New view created


- Select **File** → **Save** from Main Event List window. Views are saved with the .elv extension.

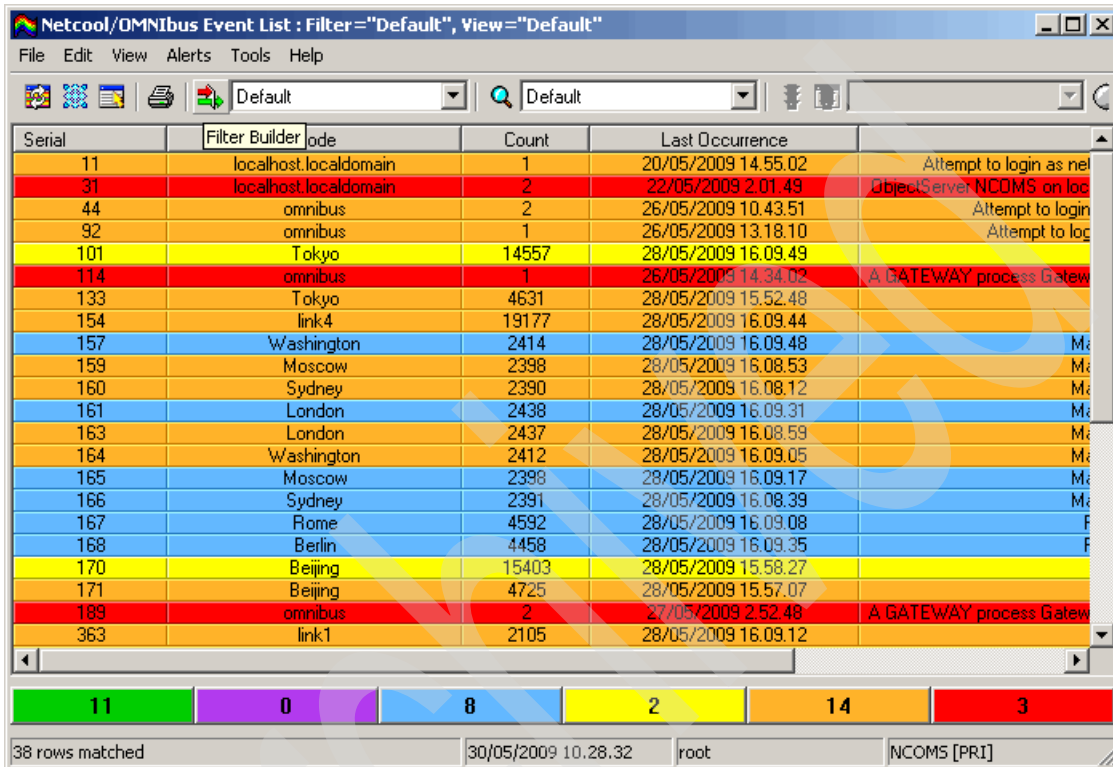
4.3.2 Filters definition

There are two types of filter that we can define in IBM Tivoli Netcool/OMNIBus: the event filter and the restriction filter. An event filter allows you to select which events you wanted to see in a view, while a restriction filter determines which events you are allowed to see for your user or group assignment.

Event filter

To define an event filter, perform these steps using the desktop client:

1. Click the Filter Builder button  on Sub-Event List, as shown in Figure 4-26.



The screenshot shows the Netcool/OMNIBus Event List window. The title bar reads "Netcool/OMNIBus Event List : Filter='Default', View='Default'". The menu bar includes File, Edit, View, Alerts, Tools, and Help. Below the menu bar is a toolbar with icons for search, filter, and other functions. The main area contains a table with the following columns: Serial, Filter Builder Code, Count, Last Occurrence, and a description. The table is filtered to show 38 rows. At the bottom of the window, there is a summary bar with colored segments representing the count of events for each filter builder code. The summary bar shows: 11 (green), 0 (purple), 8 (blue), 2 (yellow), 14 (orange), and 3 (red). Below the summary bar, it indicates "38 rows matched" and "30/05/2009 10.28.32" for the user "root" with the filter "NCOMS [PRI]".

Serial	Filter Builder Code	Count	Last Occurrence	Description
11	localhost.localdomain	1	20/05/2009 14.55.02	Attempt to login as nel
31	localhost.localdomain	2	22/05/2009 2.01.49	ObjectServer NCOMS on loc
44	omnibus	2	26/05/2009 10.43.51	Attempt to login
92	omnibus	1	26/05/2009 13.18.10	Attempt to log
101	Tokyo	14557	28/05/2009 16.09.49	
114	omnibus	1	26/05/2009 14.34.02	A GATEWAY process Gatew
133	Tokyo	4631	28/05/2009 15.52.48	
154	link4	19177	28/05/2009 16.09.44	
157	Washington	2414	28/05/2009 16.09.48	M
159	Moscow	2398	28/05/2009 16.08.53	M
160	Sydney	2390	28/05/2009 16.08.12	M
161	London	2438	28/05/2009 16.09.31	M
163	London	2437	28/05/2009 16.08.59	M
164	Washington	2412	28/05/2009 16.09.05	M
165	Moscow	2398	28/05/2009 16.09.17	M
166	Sydney	2391	28/05/2009 16.08.39	M
167	Rome	4592	28/05/2009 16.09.08	F
168	Berlin	4458	28/05/2009 16.09.35	F
170	Beijing	15403	28/05/2009 15.58.27	
171	Beijing	4725	28/05/2009 15.57.07	
189	omnibus	2	27/05/2009 2.52.48	A GATEWAY process Gatew
363	link1	2105	28/05/2009 16.09.12	

Summary bar: 11 (green), 0 (purple), 8 (blue), 2 (yellow), 14 (orange), 3 (red)

38 rows matched | 30/05/2009 10.28.32 | root | NCOMS [PRI]

Figure 4-26 Filter builder button window

2. In the Filter Builder window shown in Figure 4-27, select **File** → **New** and:
 - Provide the filter name.
 - Click the add button (+)
 - Select **Severity** from the Column drop-down menu.
 - Select **Greater than or Equal to** from the Operator drop-down menu.
 - Select **Minor (#)** from the Value drop-down menu.

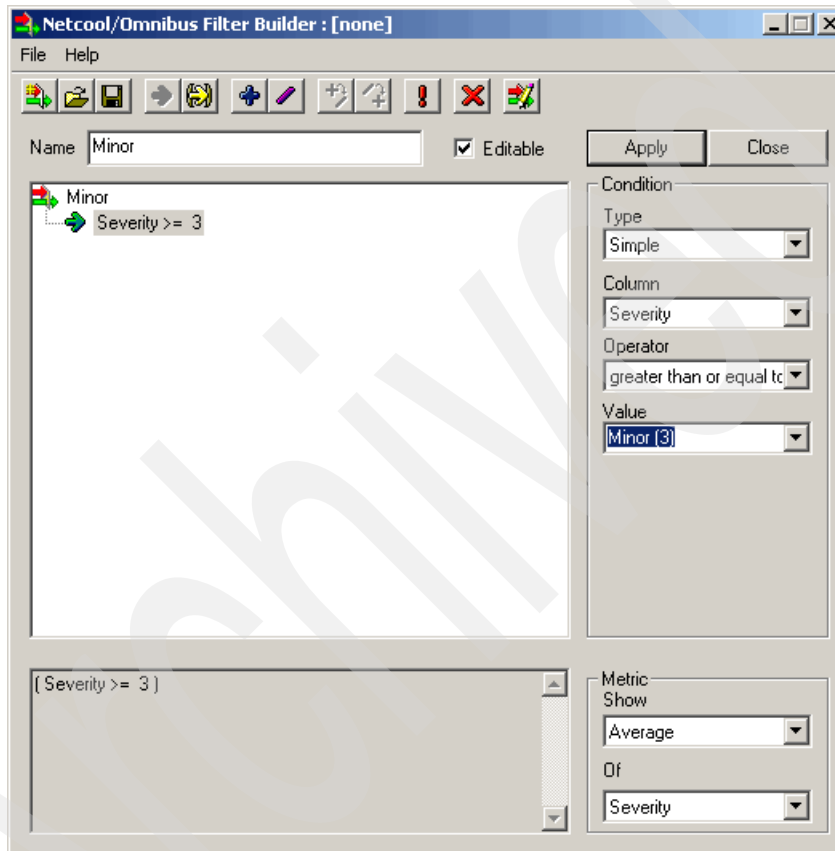


Figure 4-27 Filter Builder window

3. Click **Apply** and then **Close**.
4. Select **File** → **Save** from the Main Event List window.

Restriction filter

To define a restriction filter, perform the following steps:

1. From the main IBM Tivoli Netcool/OMNibus Administrator window, select **Users** → **Restriction Filters**, as shown in Figure 4-28.

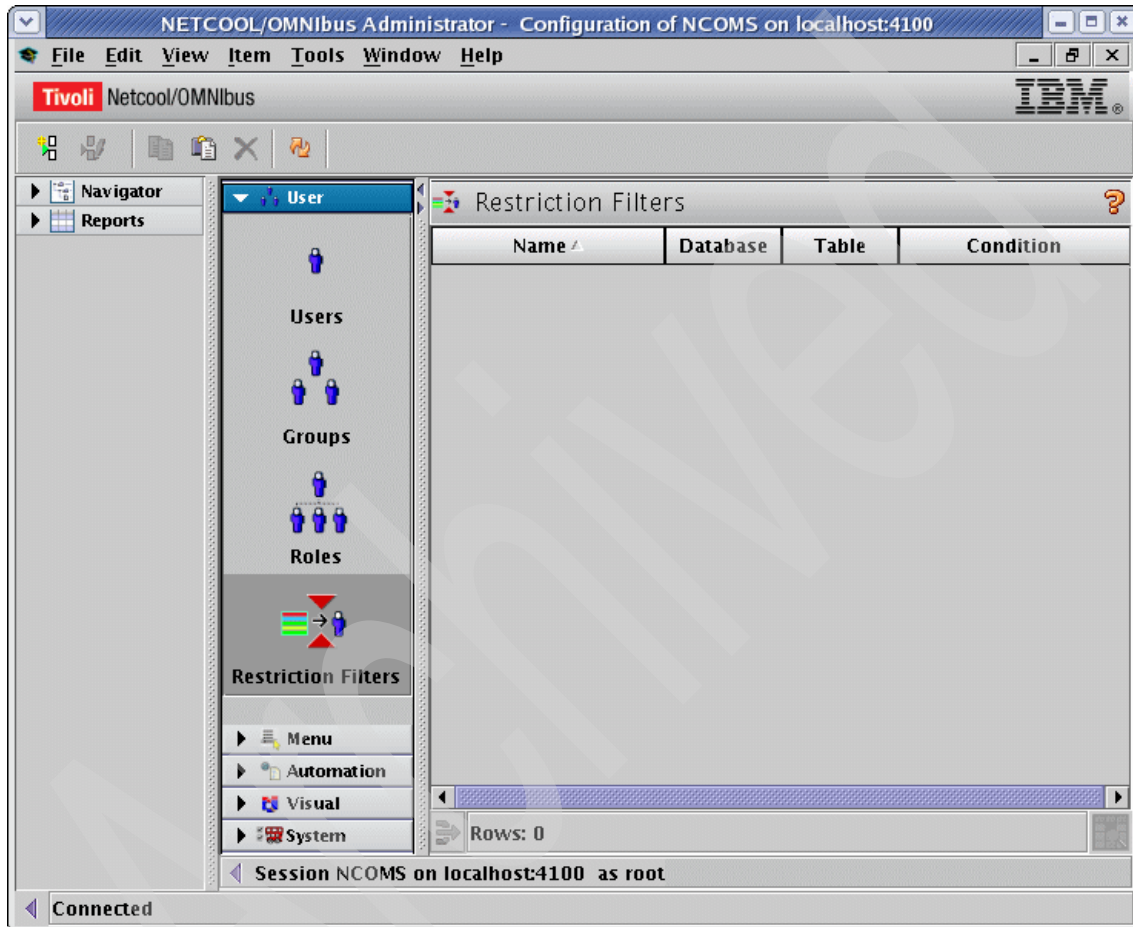



Figure 4-28 Restriction filter definition

2. Select Add Restriction Filter  from the toolbar. The creation window is shown in Figure 4-29. Assign a name to the filter and enter the filter criteria.

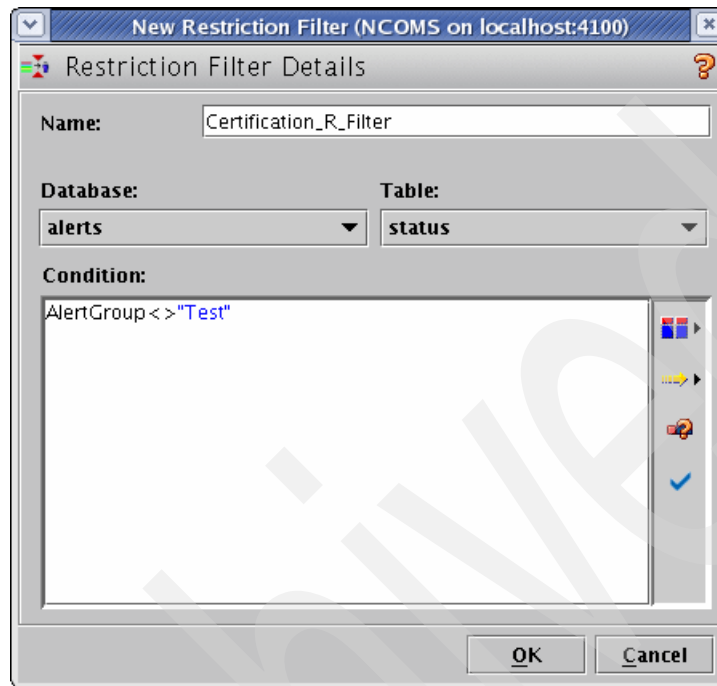


Figure 4-29 Restriction Filter creation


3. Click **OK**.

4.3.3 Menu creation

You can also modify, add, and remove menu items from the IBM Tivoli Netcool/OMNIbus display. Menu items are defined under the following items:

- ▶ Event context menu
- ▶ Main event list menu bar
- ▶ Sub-event list menu bar
- ▶ Conductor menu bar

Perform the following steps:

1. From the Menu tab, select Add New Item () within the Configuration Manager GUI, as shown in Figure 4-30.

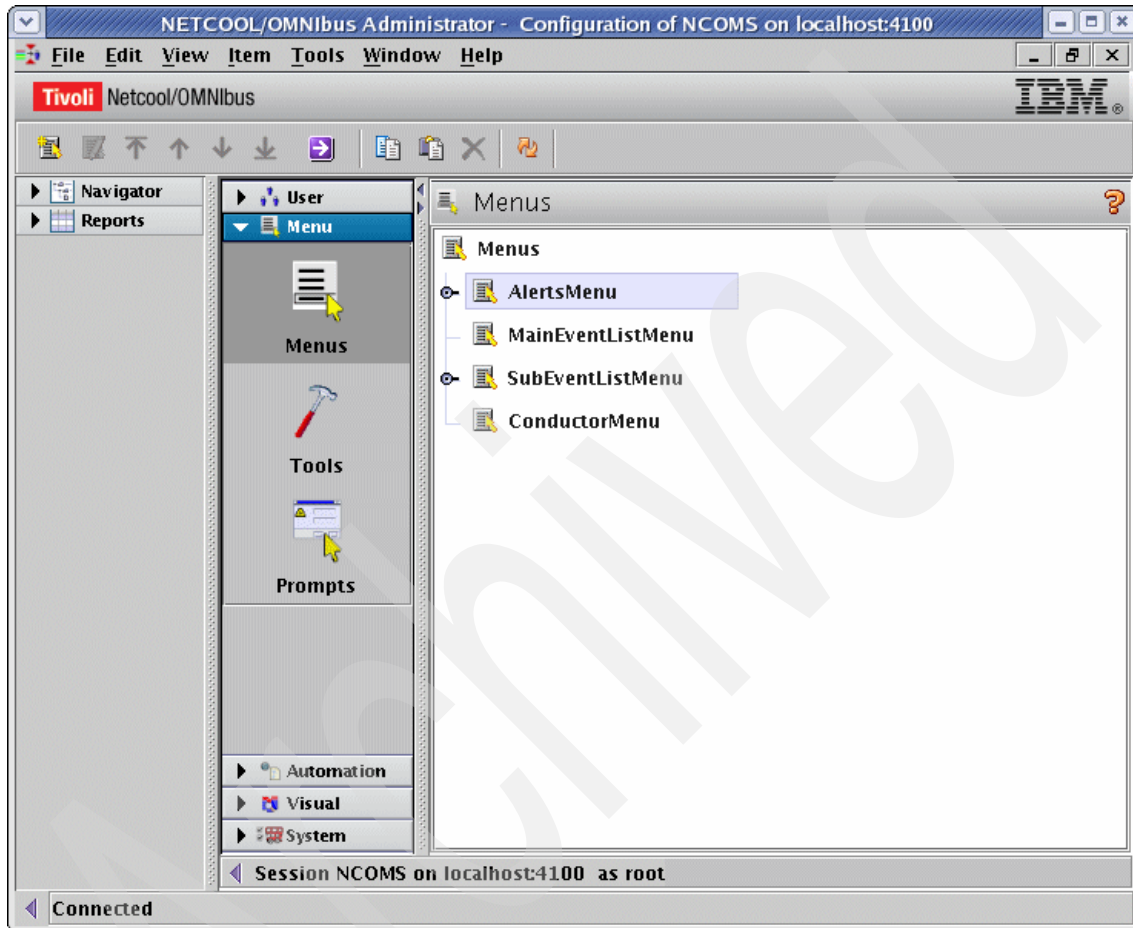


Figure 4-30 Administrator console Menu tab

- From Figure 4-31, complete the Menu Item Type, Tool, and Title fields, and then click **OK**.

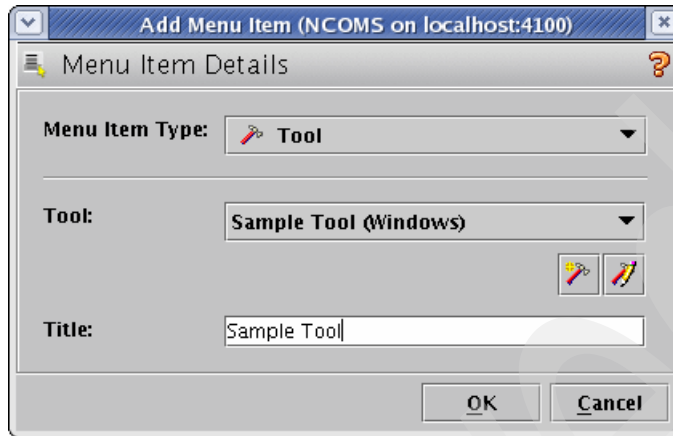


Figure 4-31 Menu definition

3. Check that the new item has been created in the administrator console, as shown in Figure 4-32.

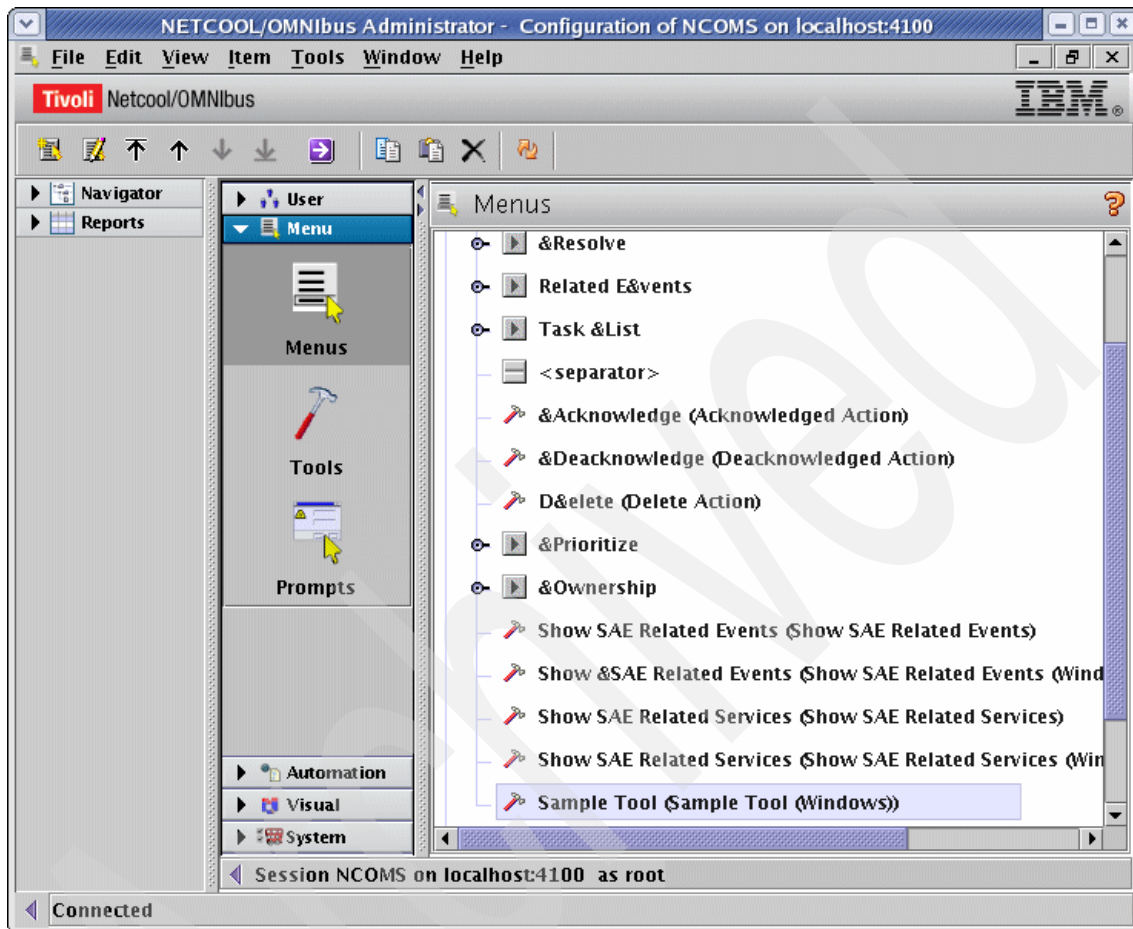


Figure 4-32 Administrator console Menu tab

4. Check that the new item has been created in the active event list (AEL), as shown in Figure 4-33.

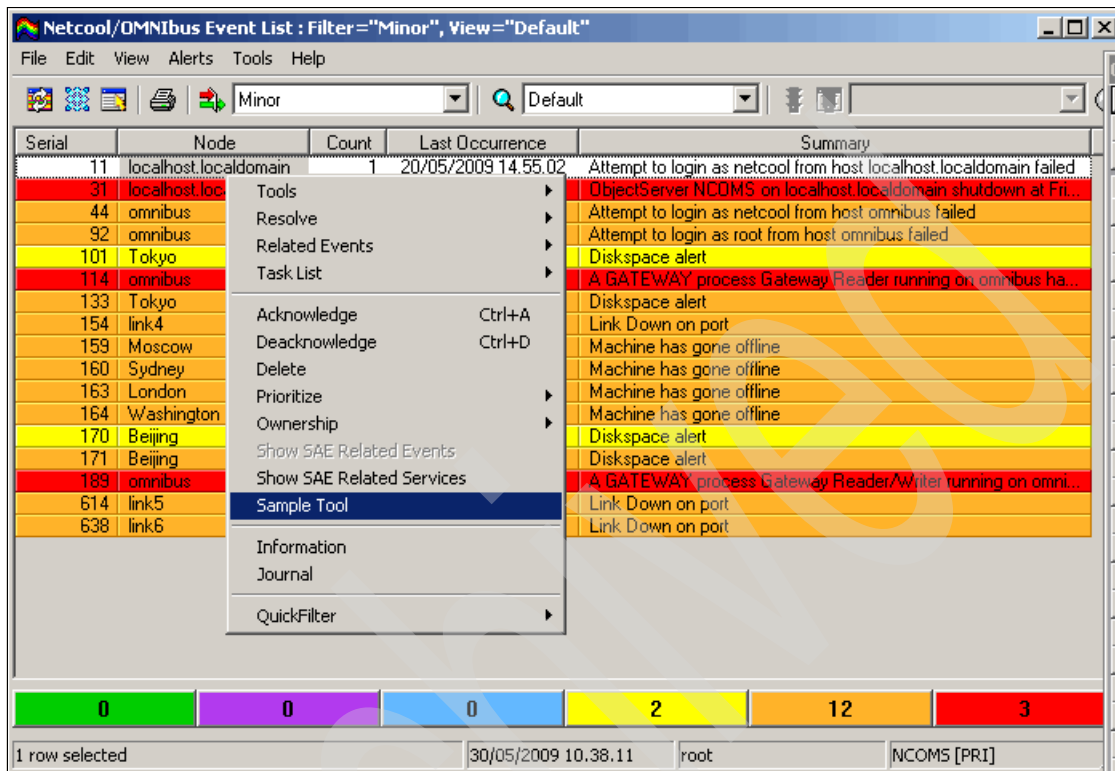


Figure 4-33 Event list tools

4.3.4 Database field definition

From IBM Tivoli Netcool/OMNIBus Administrator GUI (nco_config), you can create a new database field. Perform the following steps:

1. Select the **System** tab from the Administrator GUI.
2. Select **Databases**.
3. Expand **Databases** → **alerts** → **Status**, as shown in Figure 4-34 on page 129

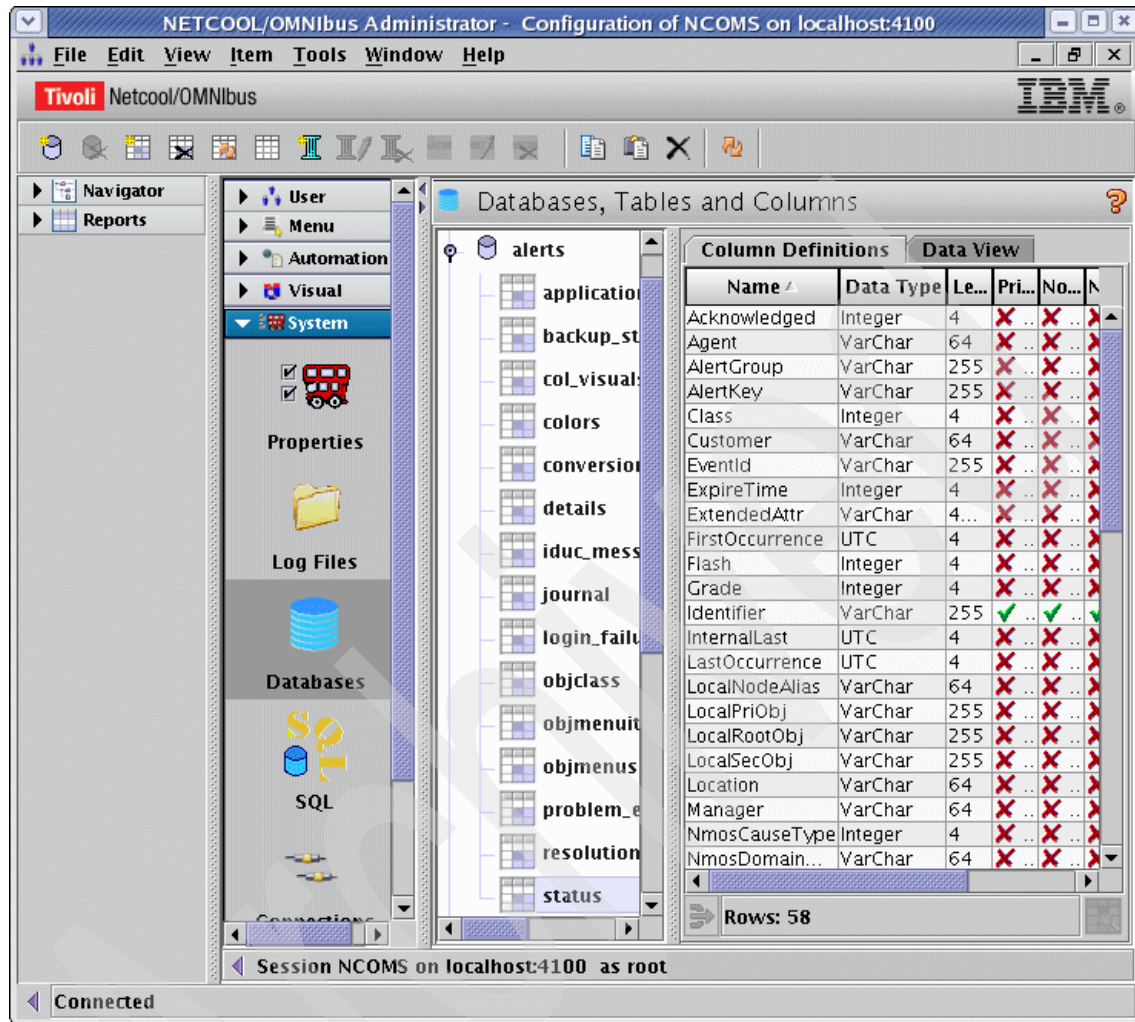



Figure 4-34 Administrator console system console

4. Select the Add Column () icon from the menu. The Add Column window is shown in Figure 4-35. Enter the Column Name and Data Type data and specify any additional attributes.

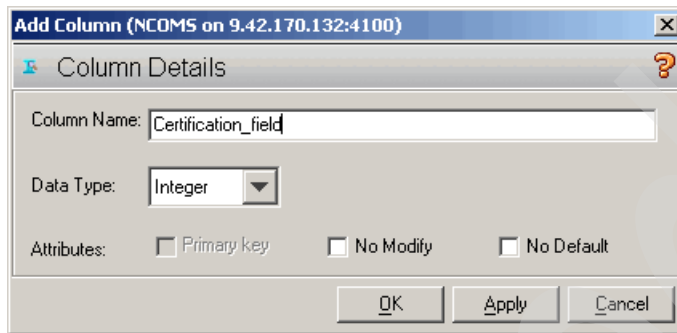


Figure 4-35 New column window

- Click **OK** and check that the column has been created correctly in the administrator console, as shown in Figure 4-36

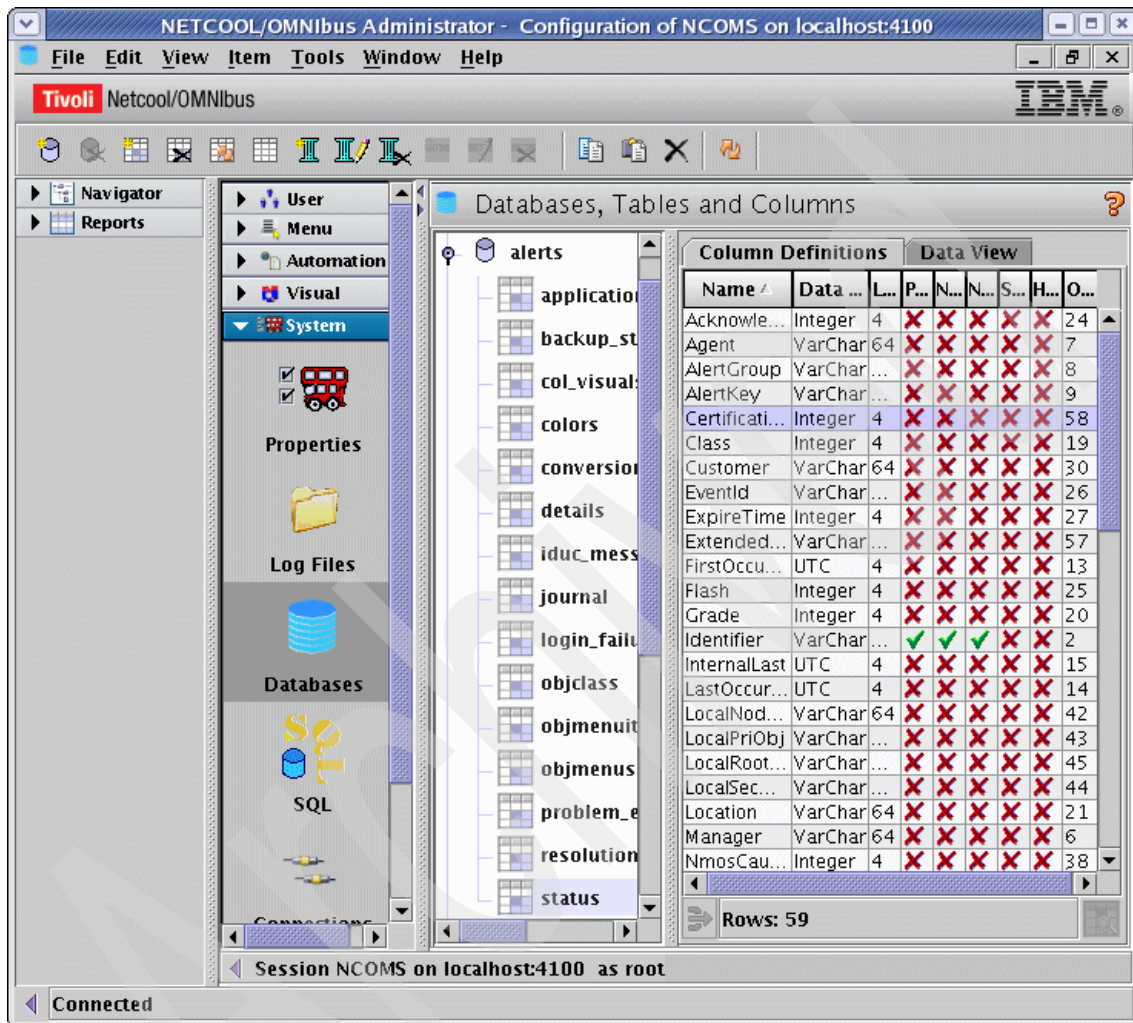


Figure 4-36 Administrator console column added

- Check that the column has been created on the event list with the default view, as shown in Figure 4-37. The field can then be used in View Builder, Filter Builder, and probe rules file.

Netcool/OMNIBus Event List : Filter="Minor", View="Default"

File Edit View Alerts Tools Help

Minor Default

Certification_field	Serial	Node	Count	Last Occurrence	Summary
0	11	localhost.localdomain	1	20/05/2009 14.55.02	Attempt to login as netcool from host localho
0	31	localhost.localdomain	2	22/05/2009 2.01.49	ObjectServer NCOMS on localhost.localdom
0	44	omnibus	2	26/05/2009 10.43.51	Attempt to login as netcool from host omnibu
0	92	omnibus	1	26/05/2009 13.18.10	Attempt to login as root from host omnibus fai
0	101	Tokyo	14700	28/05/2009 16.17.52	Diskspace alert
0	114	omnibus	1	28/05/2009 14.34.02	A GATEWAY process Gateway Reader runn
0	133	Tokyo	4672	28/05/2009 16.18.15	Diskspace alert
0	154	link4	19236	28/05/2009 16.18.17	Link Down on port
0	159	Moscow	2406	28/05/2009 16.17.47	Machine has gone offline
0	160	Sydney	2399	28/05/2009 16.18.04	Machine has gone offline
0	163	London	2446	28/05/2009 16.17.15	Machine has gone offline
0	164	Washington	2422	28/05/2009 16.18.05	Machine has gone offline
0	170	Beijing	15403	28/05/2009 15.58.27	Diskspace alert
0	171	Beijing	4725	28/05/2009 15.57.07	Diskspace alert
0	189	omnibus	2	27/05/2009 2.62.48	A GATEWAY process Gateway Reader/Win
0	363	link1	2124	28/05/2009 16.18.19	Link Down on port
0	614	link5	38	28/05/2009 16.18.16	Link Down on port
0	632	link3	27	28/05/2009 16.17.54	Link Down on port
0	633	link2	42	28/05/2009 16.17.28	Link Down on port
0	638	link6	17	28/05/2009 16.17.30	Link Down on port


0 0 0 2 15 3

20 rows matched 30/05/2009 10.43.34 root NCOMS [PRI]

Figure 4-37 Event list column

4.3.5 Tool definition

Creating a new tool allows you to define an additional program that you can invoke from the Administration GUI. Perform the following steps:

1. Within the Tools menu in Figure 4-38, click the Add Tool  icon.

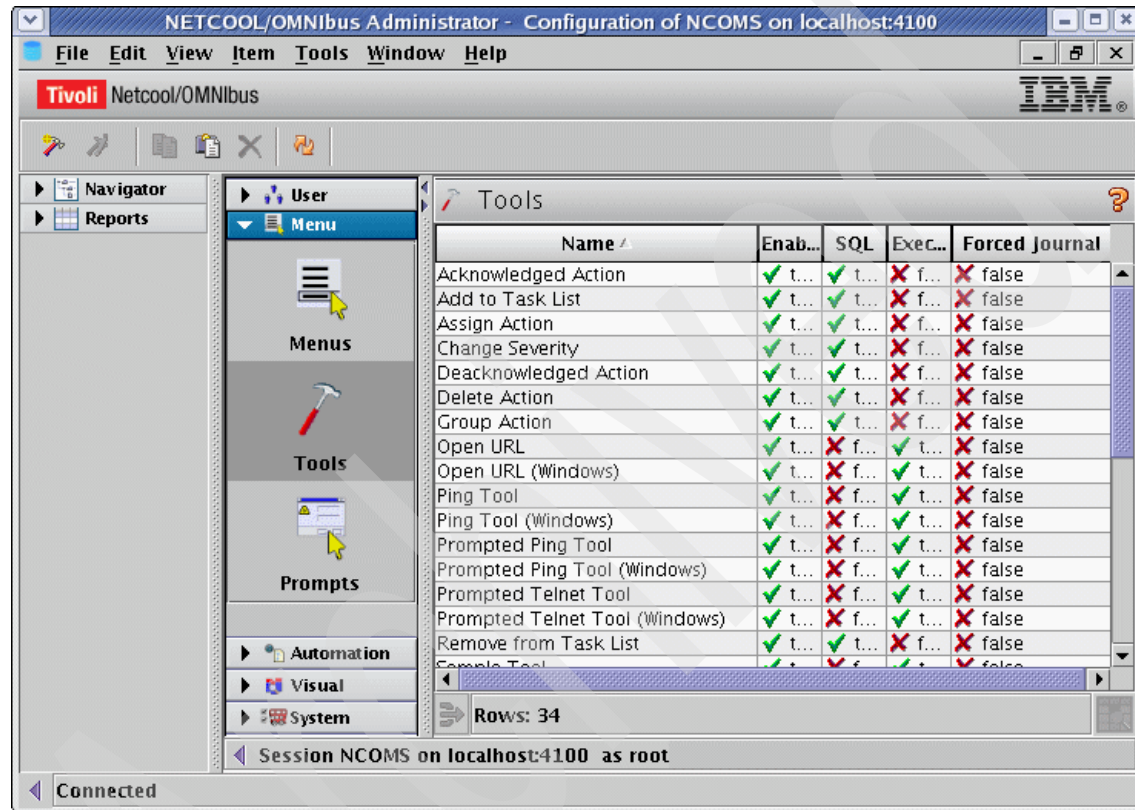


Figure 4-38 Administrator console Menu tab

2. In the New Tool window, enter a Name. You can have a tool that executes SQL statements against the ObjectServer database command, as shown in Figure 4-39.

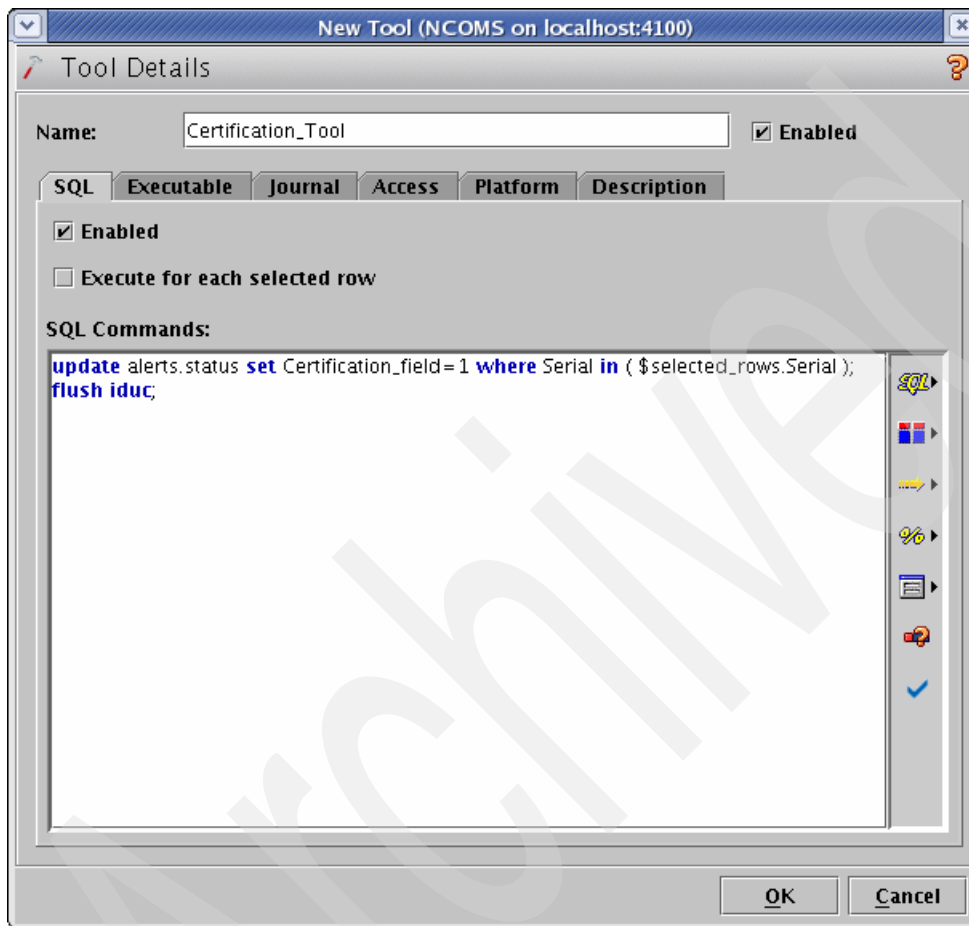


Figure 4-39 Tool creation window

3. If you want to run an executable, enter the executable name in the Executable tab, as shown in Figure 4-40.

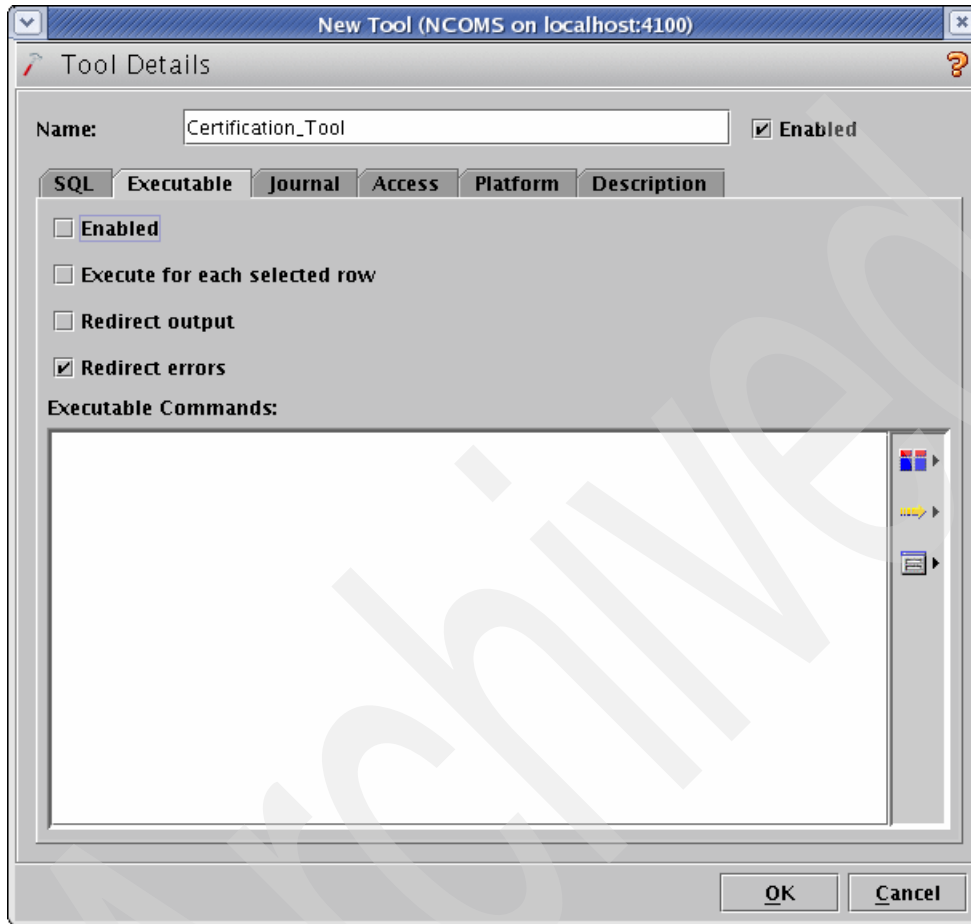


Figure 4-40 Tool creation window

When you use the executable, the executable has to be in the same path as all users and resides must be checked on the same platform. You must check the platform support on the Platform tab, as shown in Figure 4-41.

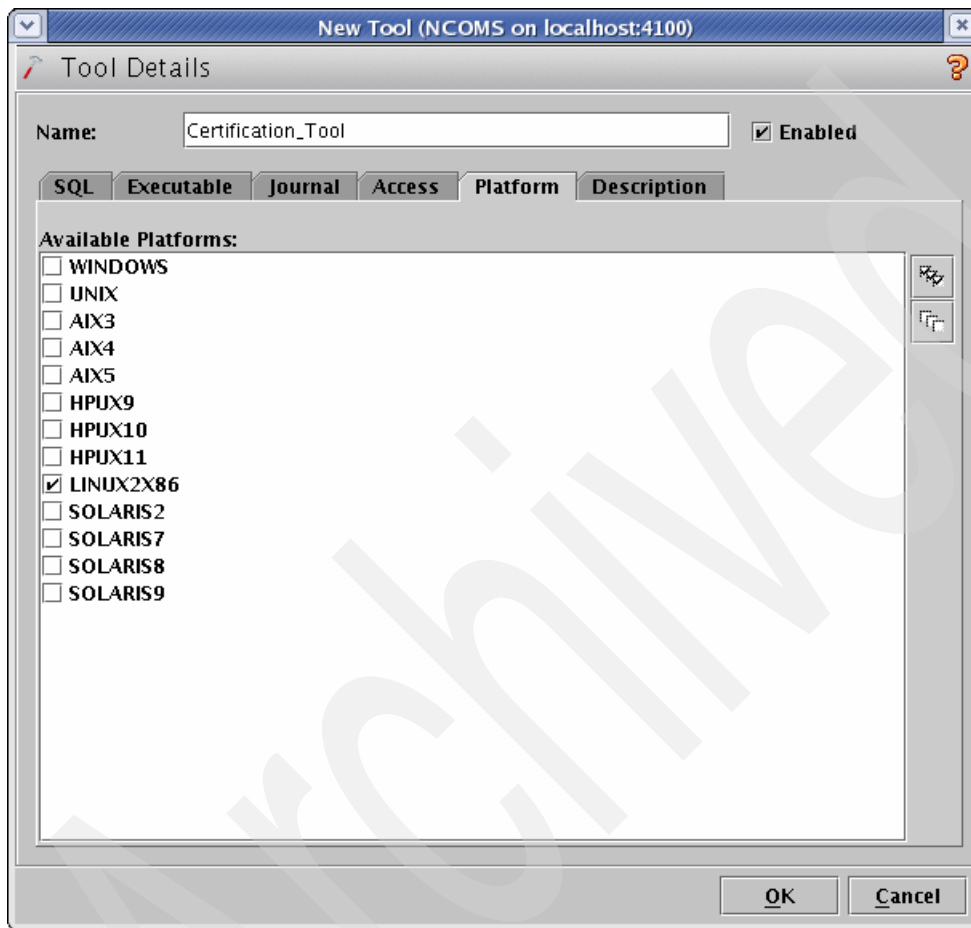


Figure 4-41 Tool creation window

4. The tools' access rights can be configured from the Access tab, as shown in Figure 4-42 on page 137.

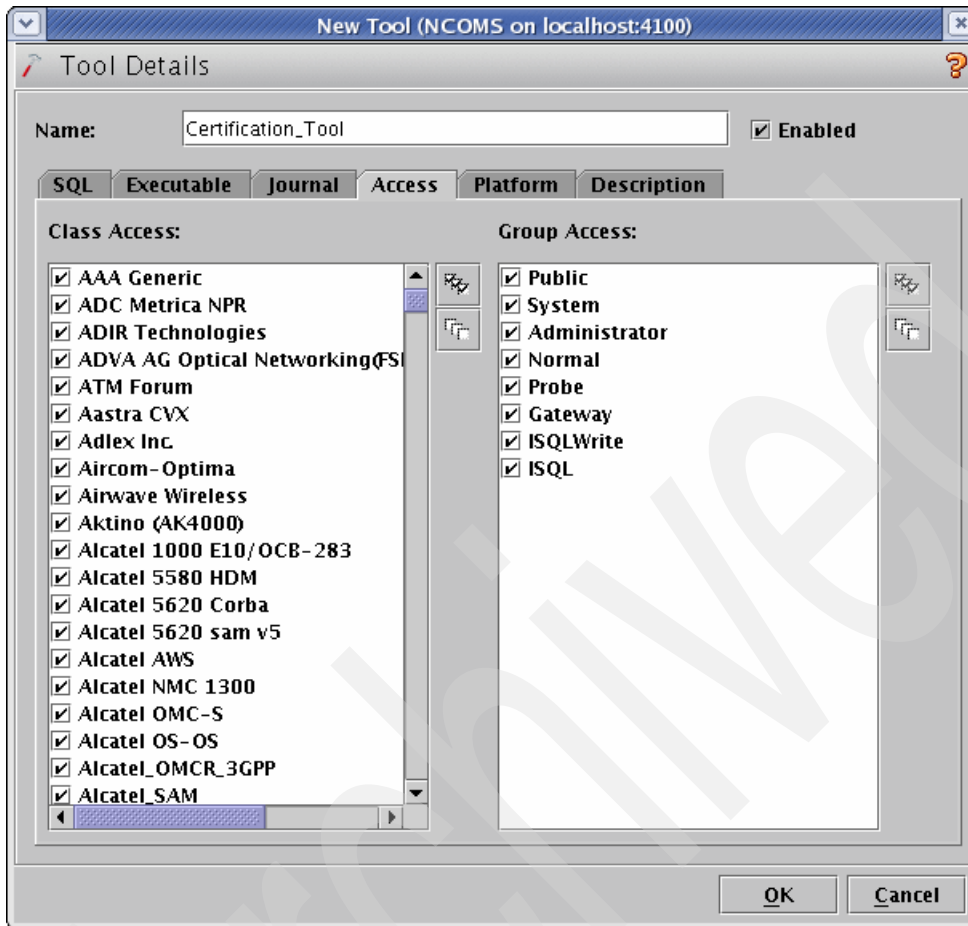


Figure 4-42 Tool creation window

5. Click **OK** and check that the tool has been created correctly in the menu window.

- There is a way to collect operator input from the tools before running the executable. This is called a prompt. You define the prompt by selecting **Menu** → **Prompt**, as shown in Figure 4-43.

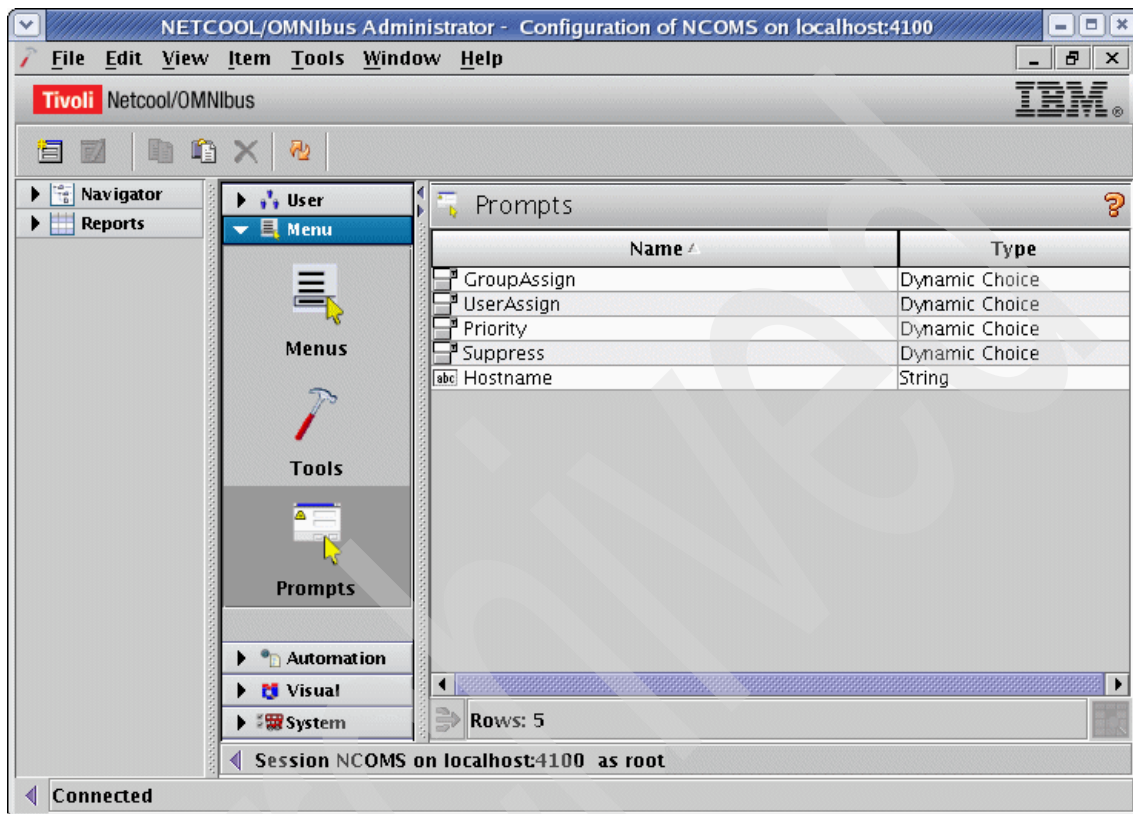


Figure 4-43 Prompt

- The prompt definition window is shown in Figure 4-44 on page 139.

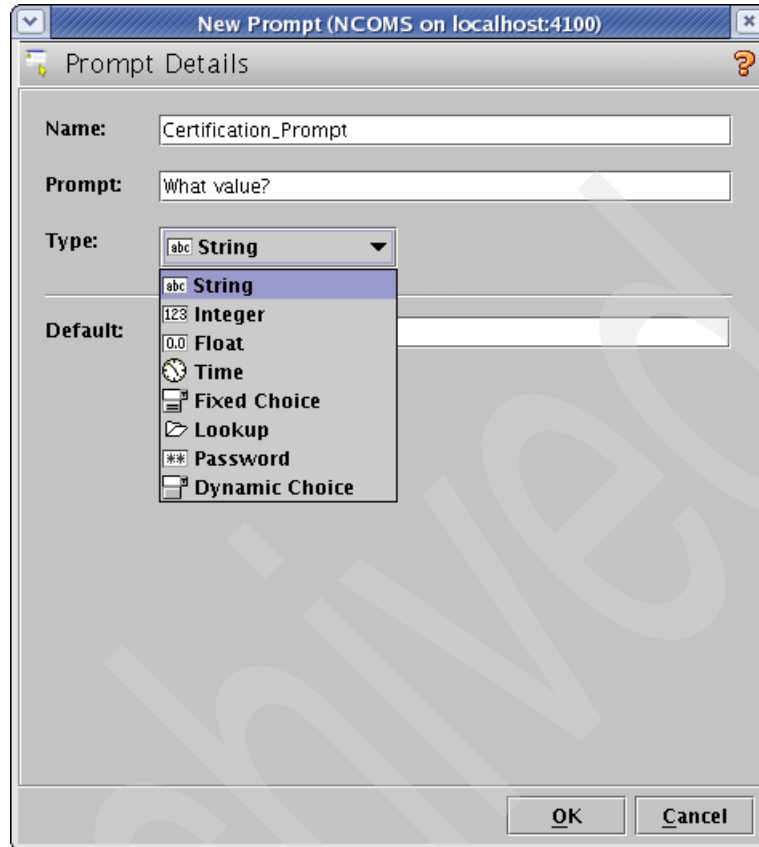


Figure 4-44 Prompt definition

There are several different prompt types:

String	Gets a string value.
Integer	Gets an integer value.
Float	Gets a floating point value.
Time	Gets a time stamp information.
Fixed Choice	Gets a choice of a predefined selection.
Lookup	Gets value from a file.
Password	Gets a password field.
Dynamic Choice	Gets the choices from a database table entries.

8. The prompt can be included in the tool definition as `$prompt.<name>` to get the value from the user input.

4.3.6 Trigger creation

Triggers are accessed by selecting **Menu** → **Trigger**, as shown in Figure 4-45.

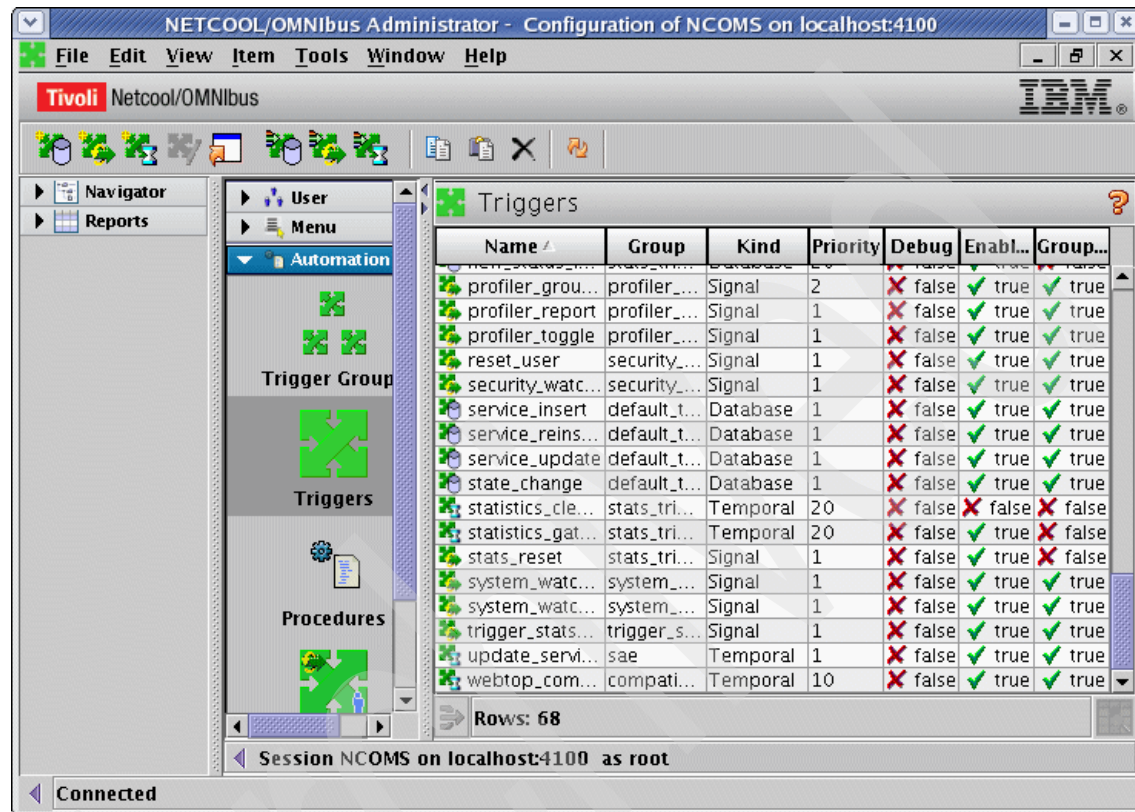


Figure 4-45 Triggers

There are several different types of triggers:

- ▶ A database trigger (🗄️) allows you to run a program whenever there are changes on the database. A typical trigger would be run upon the insertion, update, or deletion of a database row. A trigger on the alerts table would allow you to monitor whether an event is received or updated.
- ▶ A temporal trigger (🕒), which is executed on a predefined frequency.
- ▶ A signal trigger (🌿) fires when a system or user-defined signal is raised. System signals are raised spontaneously by the ObjectServer when it detects changes to the system. Signals are identified as %signal.<signalname>.

To define a trigger (Figure 4-46), you need the following definitions:

- ▶ Setting tab: Defines what event that this trigger would be attached to, such as:
 - Database trigger: Defines the tables and whether it is a deletion, insertion, reinsertion, or update.
 - Signal trigger: Which signal is defined for this trigger.
 - Temporal trigger: Specifies the execution frequency.
- ▶ When tab: Defines the database condition that has to be fulfilled for the trigger to run.
- ▶ Action tab: Defines the SQL procedure to be executed.

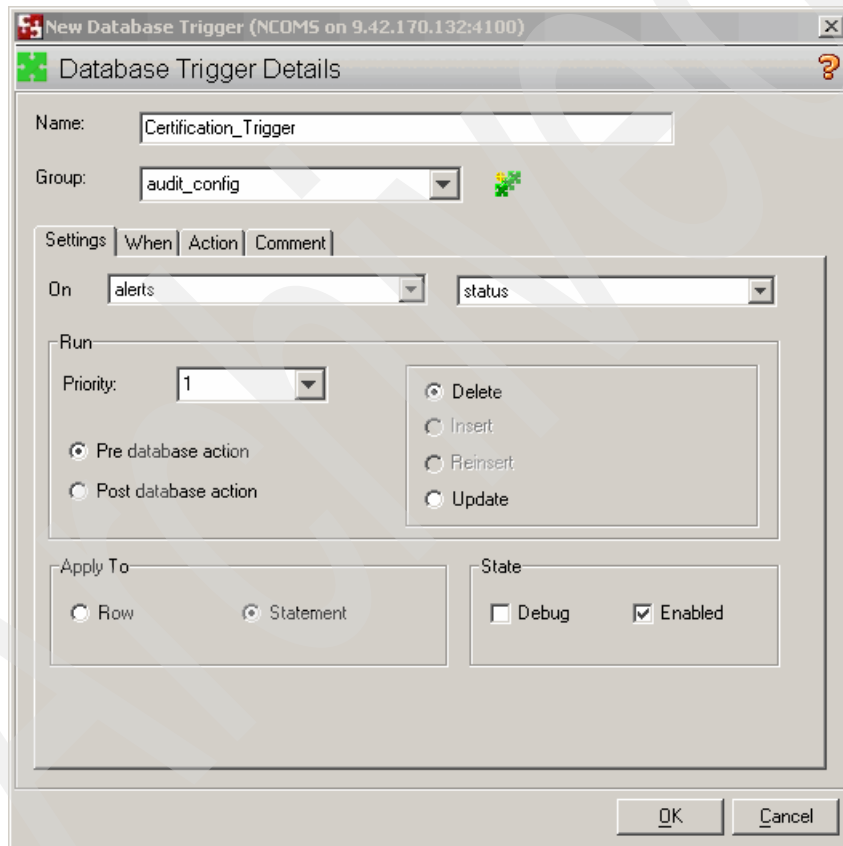


Figure 4-46 Database trigger creation settings tab

Triggers are defined in trigger groups. The group allows triggers to be enabled or disabled together. To do this in a command line using `nco_sql` or `isql`, run:

```
ALTER TRIGGER GROUP <group> SET ENABLED (TRUE | FALSE)
```

In addition to using the configuration GUI, the triggers can also be defined using an SQL command. The SQL command syntax is shown in Example 4-14.

Example 4-14 SQL statement to define a trigger

```
CREATE [ OR REPLACE ] TRIGGER triggername GROUP trgroup { BEFORE |
AFTER } { INSERT | UPDATE | DELETE | REINSERT } ON
database_name.table_name FOR EACH { ROW | STATEMENT }
BEGIN
-- comments
trigger_action
END;
```

Some important statements in a trigger are:

- ▶ A procedure can also be called by a trigger by using an EXECUTE SQL command.
- ▶ You can break the execution in a loop by using a BREAK command.

4.3.7 Class creation

Alerts in IBM Tivoli Netcool/OMNIBus are defined in different classes to ease processing for a certain class. New classes can be added with reference to an existing class. Perform the following steps:

1. In the IBM Tivoli Netcool/OMNIBus Administration window, select **Visual** → **Classes**, as shown in Figure 4-47. Click the Add Class (🎓) icon.

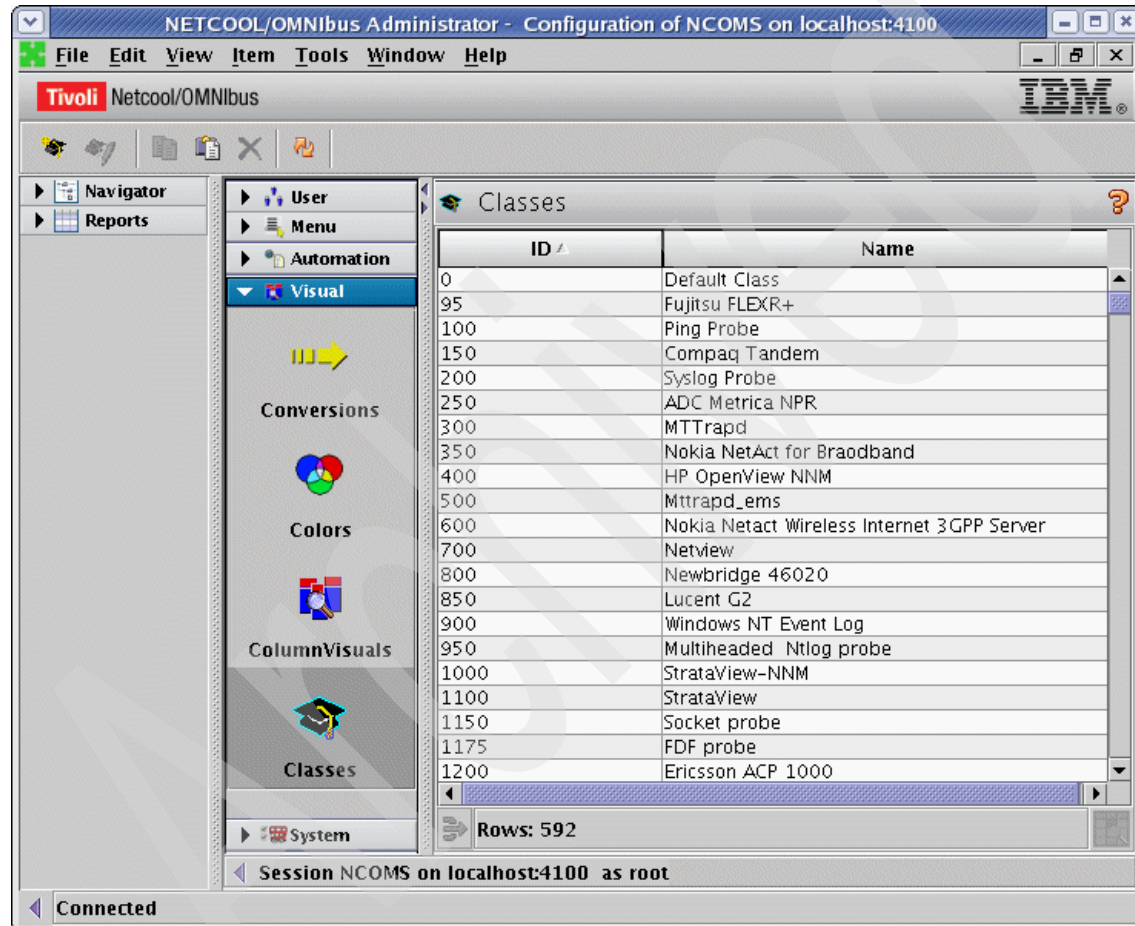


Figure 4-47 Administrator console Visual window

2. In the new class window shown in Figure 4-48, perform the following steps:
 - a. Assign an identifier.
 - b. Name the class.
 - c. Click **OK**.

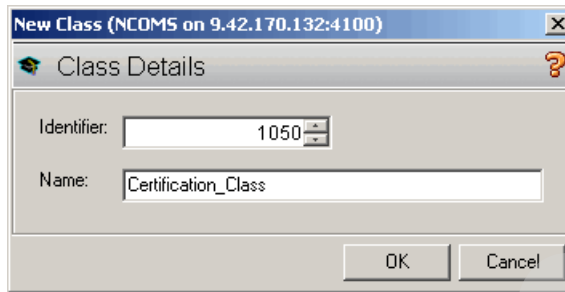


Figure 4-48 Add Class window

4.4 Probe configuration

Probes connect to an event source, detect and acquire event data, and forward the data to the ObjectServer as alerts. Probes use the logic specified in a rules file to manipulate the event elements before converting them into fields of an alert in the ObjectServer alerts.status table. Probe configuration includes modifying the following items:

- ▶ 4.4.1, “Probe configuration sample” on page 144
- ▶ 4.4.2, “Probe property” on page 146
- ▶ 4.4.3, “Probe rule file” on page 150

4.4.1 Probe configuration sample

You must configure each probe configuration to connect to the proper ObjectServer. The ObjectServer is pointed from the Server property of the probe property file. The example in this section configures the simnet probe.

Perform the following steps:

1. Open the Probe properties file. For simnet, it is called `simnet.props` and is found under `$OMNIHOME/probes/<arch>`, where `arch` refers to the machine architecture, which is `linux2x86` for our Linux system.
2. Set the server property to the name of the ObjectServer, as shown in Example 4-15 on page 145. Other probe properties are discussed in 4.4.2, “Probe property” on page 146.

Example 4-15 simnet.props

```
[netcool@omnibus linux2x86]$ cat simnet.props | grep Server
Server : 'NCOMS'
```

3. Modify the rules file, `simnet.rules`, to include the host system name in the manager field, as shown in Example 4-16. Rules are discussed at greater length in 4.4.3, “Probe rule file” on page 150.

Example 4-16 simnet.rules

```
[netcool@omnibus linux2x86]# cat simnet.rules
{
    @Manager      = "Simnet Probe" + ":" + $Node
    @Class        = 3300
    @Node         = $Node
    @Agent        = $Agent
    @AlertGroup   = $Group
    @Summary      = $Summary
    @Severity     = $Severity
}
```

4. Start the probe by using the probe executable. The `simnet` probe is started by running the `nco_p_simnet` command under `$OMNIHOME/probes/`. Some of the arguments are:

-server Overrides the props file.
-errorlevel Defines the debugging message level.

5. Check the process list to verify that the probe is running and run `ps -ef` and `grep` with the option `simnet`, as shown in Example 4-17.

Example 4-17 Verifying the probe is running

```
[netcool@omnibus linux2x86]$ ps -ef | grep simnet
root      11124 10751  0 19:15 pts/2    00:00:00 ./nco_p_simnet
netcool   11188 10572  0 19:17 pts/1    00:00:00 grep simnet
```

- Launch the event list and verify the probe events in the ObjectServer and verify that the events parsed correctly, as shown in Figure 4-49.

Manager	Node	Alert Group	Summary
Simnet Probe:Beijing	Beijing	Stats	Diskspace alert
Simnet Probe:link4	link4	Link	Link Down on port
Simnet Probe:link6	link6	Link	Link Down on port
Simnet Probe:Washington	Washington	Systems	Machine has gone offline
Simnet Probe:	link1	Link	Link Down on port
Simnet Probe:link2	link2	Link	Link Down on port
Simnet Probe:Moscow	Moscow	Systems	Machine has gone offline
Simnet Probe:London	London	Systems	Machine has gone offline
Simnet Probe:Beijing	Beijing	Stats	Diskspace alert
Simnet Probe:Rome	Rome	Link	Port failure : port reset
Simnet Probe:Washington	Washington	Systems	Machine has gone online
Simnet Probe:Moscow	Moscow	Systems	Machine has gone online
Simnet Probe:London	London	Systems	Machine has gone online
Simnet Probe:Berlin	Berlin	Link	Port failure : port reset
Simnet Probe:Sydney	Sydney	Systems	Machine has gone online
Simnet Probe:link4	link4	Link	Link Up on port

0 0 6 1 8

16 row(s) matched.

Figure 4-49 IBM Tivoli Netcool/OMNIBus, Event List

- Check for errors in the log file `$NCHOME/omnibus/log/simnet.log`.

4.4.2 Probe property

The probe property file contains some common parameters and also probe specific parameters. The common parameters are discussed in Table 4-3 on page 147.

Table 4-3 Probe property parameters

Property	Description
AuthPassword string	User name and password to authenticate the probe to ObjectServer for running the probe in secure mode.
AuthUserName string	
AutoSAF 0 1	Specifies whether automatic store-and-forward mode is enabled. It is not enabled by default. It must be enabled for failover to work.
BeatInterval integer	Specifies the heartbeat interval for peer-to-peer failover. The default is 2 seconds.
Buffering 0 1	Specifies whether buffering is used when sending alerts to the ObjectServer. Alerts order is preserved for each table, but not across tables.
BufferSize integer	Specifies the number of alerts the probe buffers. The default is 10.
LogFilePoolSize integer	Specifies the number of log files to use. The pool size can range from 2 to 99. The default is 10. (Windows only)
LogFileUsePool 0 1	Specifies whether to use a pool of log files for logging messages. If set to 1, the logging system opens the number of files specified for the pool at startup, and keeps them open for the duration of its run. When set to 0, the default probename.log file is renamed probename.log_OLD and a new log file is started when the maximum size is reached. The default is 0. (Windows only)
LogFileUseStdErr 0 1	Specifies whether to use stderr as an output stream for logging messages. The default is 1, which causes messages to be logged to the console only if the probe was run from the command line.
LookupTableMode integer	Specifies how table lookups are performed. It can be set to 1, 2, or 3. The default is 3 (check the number of column first).
Manager string	Specifies the value of the Manager field for the alert. The default value is determined by the probe.
MaxLogFileSize integer	Specifies the maximum size that the log file can grow to, in bytes. The default is 1 MB.
MaxRawFileSize integer	Specifies the maximum size of the raw capture file, in KB. The default is unlimited (-1).
MaxSAFFileSize integer	Specifies the maximum size the store-and-forward file can grow to, in bytes. The default is 1 MB.
MessageLevel string	Specifies the message logging level. Possible values are debug, info, warn, error, and fatal. The default level is warn.

Property	Description
MessageLog string	Specifies where messages are logged. MessageLog can also be set to stdout or stderr. The default is \$OMNIHOME/log/probename.log.
Mode string	Specifies the role of the instance of the probe in a peer-to-peer failover relationship. The mode can be master/slave/standard. The default is standard.
MsgDailyLog 0 1	Specifies whether daily logging is enabled. By default, the daily backup of log files is not enabled (0). Because the time is checked regularly, when MsgDailyLog is set, there is a slight reduction in performance.
MsgTimeLog string	Specifies the time after which the daily log is created. The default is 0000 (midnight). If MsgDailyLog set to 0, this value is ignored.
Name string	Specifies the name of the probe. This value determines the names of the properties file, rules file, message log file, and store-and-forward file.
NetworkTimeout integer	Specifies the length of time (in seconds) that the probe can wait without a response; after this time, the connection to the ObjectServer times out. The default is 0, meaning that no timeout occurs. If a timeout occurs, the probe attempts to connect to the backup ObjectServer, identified by the ServerBackup property. If a timeout occurs and no backup ObjectServer is specified, the probe enters store-and-forward mode.
PeerHost string	Specifies the host name of the network element acting as the counterpart to this probe instance in a peer-to-peer failover relationship. The default is localhost.
PeerPort integer	Specifies the port through which the master and subordinate communicate in a peer-to-peer failover relationship. The default port is 99.
PidFile string	Specifies the name of the file that stores the process ID for the device. The default is \$OMNIHOME/var/name.pid, where name is the name of the probe and pid is the process ID.
PollServer integer	The frequency in seconds at which the probe polls for the return of the primary ObjectServer. If connected to a backup ObjectServer because failover occurred, a probe periodically attempts to reconnect to the primary ObjectServer. The default is 0, meaning that no polling occurs.
Props.CheckNames TRUE FALSE	When TRUE, the probe does not run if any specified property is invalid. The default is TRUE.
PropsFile string	Specifies the name of the properties file. The default is \$OMNIHOME/probes/arch/name.props, where name is the name of the probe and arch represents the operating system.

Property	Description
RawCapture 0 1	Controls the raw capture mode. Raw capture mode is usually used at the request of IBM Technical Support. By default, raw capture mode is disabled (0).
RawCaptureFile string	Specifies the name of the raw capture file. The default is \$OMNIHOME/var/name.cap, where name is the name of the probe.
RawCaptureFileAppend 0 1	Specifies whether new data is appended to the existing raw capture file, instead of overwriting it. By default, the file is overwritten (0).
RetryConnectionCount integer	Specifies the number of events the probe processes in store-and-forward mode before trying to reconnect to the ObjectServer. The default is 15.
RetryConnectionTimeOut integer	Specifies the number of seconds that the probe processes events in store-and-forward mode before trying to reconnect to the ObjectServer. The default is 30.
RulesFile string	Specifies the name of the rules file. This can be a file name or Web address that specifies a rules file located on a remote server that is accessible using HTTP. The default is \$OMNIHOME/probes/arch/name.rules, where name is the name of the probe.
SAFFileName string	Specifies the name of the store-and-forward file. The default is \$OMNIHOME/var/name.store.server, where name is the name of the probe and server is the name of the target ObjectServer.
Server string	Specifies the name of the primary ObjectServer or the proxy server to which alerts are sent. The default is NCOMS.
ServerBackup string	Specifies the name of a backup ObjectServer to which the probe should connect if the primary ObjectServer connection fails. If NetworkTimeout is set, use ServerBackup to identify a backup ObjectServer.
StoreAndForward 0 1	Controls the store and forward operations. By default, store-and-forward mode is enabled (1).

4.4.3 Probe rule file

The probe rule file determines how a probe sends events to ObjectServer. A probe rules file resides in `$NCHOME/omnibus/probes/<arch>/`, and has the file type of rules. You can customize the rule with a text editor. This section discusses some sample constructs that can be used with a rule file:

- ▶ A simple assignment with the = sign can be used to transfer the result of an expression to another variable. A variable can be one of:
 - The column name is prefixed with an @ sign.
 - A temporary variable name is prefixed with a \$ sign.
 - A probe property value is prefixed with a % sign.
 - Array and tables must be declared before any processing directives.

- ▶ In a probe rules file, an array is defined with the array directive. Arrays are one dimensional. An array assignment is written as:

```
node_arr["myhost"] = "a"
```

Array values are persistent until a probe is restarted; if you force the probe to re-read the rules file by issuing a `kill -HUP` command, the array values are maintained.

- ▶ The extract statement is used to apply a regular expression to a variable or field. An example is to extract a machine name from the Summary column that is prefixed with the word Machine:. Use the expression `extract(@Summary,"^Machine:(.*)")` to parse the Summary column value and find the word Machine: and extract the remaining characters. That statement would map:

```
– Machine:test1 → test1
– Machine:sample content → sample content
```

- ▶ Regular expression basic constructs:

<code>^</code>	Start of string.
<code>\$</code>	End of string.
<code>[]</code>	A value list, which can be a list or range of values [0-9], which is the same as [01-567-9].
<code>.</code>	Any characters.
<code>*</code>	Zero or more characters of the previous item.
<code>+</code>	One or more characters of the previous item.
<code>?</code>	Any single character.
<code>\</code>	Escape character for the next character.
<code>()</code>	Section that would represent the returned value.

- ▶ The switch construct is useful to branch a multiple decision for mapping a value from one variable to a different action. A sample switch statement is shown in Example 4-18. It changes the summary field for events coming from nodes London and Moscow.

Example 4-18 Switch example

```
switch(@Node)
{
  case "London":
    @Summary = "Sample " + @Node + " bridge test"
  case "Moscow":
    @Summary = "Switch rule test for node: " +@Node
  default:
    @Summary = @Summary
}
```

- ▶ You can use the include statement to include the content of another rule file in the current block. In Example 4-19, we process every event which has a Node value equal to Rome. The entire content of rome.rules would be substituted for the action block for the if statement. The included rules file cannot have table or array definitions, as they would make the table or array defined after a processing directive.

Example 4-19 Include sample

```
if(match($Node, "Rome"))
{ include "/opt/netcool/omnibus/probes/linux2x86/rome.rules" }
```

- ▶ For deduplication, so that a field gets updated upon deduplication, you can use the update function. The syntax of the update function is:
update(field [, TRUE|FALSE]);
TRUE represents that update is enabled, which is the default.
- ▶ The matching of a string is performed using the regmatch function. It has the format of regmatch(field, regex), which matches the field against the regular expression. Example 4-20 discards an event when the Summary field contains the word test.

Example 4-20 Discard and regmatch example

```
if (regmatch($Summary, ".*test.*"))
{ discard }
```

- ▶ A lookup table is used to look up a value from a lookup file. For example, the lookup file shown in Example 4-21 is based on a node name.

Example 4-21 Lookup table

```
London "Finance"  
Rome   "Finance"  
Moscow "Technical"  
Berlin "Technical"
```

You can reference the external lookup table file, which is similar to Example 4-22. It assigns the department code depending on the content of the Node field. The London and Rome nodes map to the Finance department while the Moscow and Berlin nodes map to the Technical department.

Example 4-22 rules file table definition

```
table dept="/opt/netcool/omnibus/probes/linux2x86/cert"  
.  
.  
.  
@Department=lookup(@Node,dept)
```

- ▶ To turn on probe detail, use the `details($*)` statement.
- ▶ The `registertarget` function registers one or more ObjectServers, and the corresponding tables, to which you might want to send alerts. You can use the `setdefaulttarget` function to change the default ObjectServer to which alerts are sent when a target is not specified. The format for the `registertarget` function is:

```
registertarget(server_name, backupserver_name, alerts_table [,  
details_table ] )
```

- ▶ If you want to update the probe rules file without stopping the probe, you can do so by sending a SIGHUP to the probe process ID. This is demonstrated in Example 4-23.

Example 4-23 Refreshing the probe

```
[netcool@omnibus probes]$ ps -ef | grep simnet  
netcool  2177 29247  0 00:27 pts/2    00:00:02 ./nco_p_simnet  
netcool  3701 29247  0 00:53 pts/2    00:00:00 grep simnet  
[netcool@omnibus probes]$ kill -HUP 2177
```

To use the syntax of a custom rules file, install the IBM Tivoli Netcool/OMNIBus Probe Rules Syntax Checker (part number C1BX7EN). The installation of the Probe Rules Syntax Checker is shown in Example 4-24 on page 153.

Example 4-24 Installing Probe Rules Syntax Checker

```
[netcool@omnibus tmp]$ unzip C1BX5EN.zip
Archive:  C1BX5EN.zip
  extracting: omnibus-3.x-linux2x86-probe-nco-p-syntax-3_0.tar.Z
  inflating: omnibus-3.x-linux2x86-probe-nco-p-syntax-3_0.README
[netcool@omnibus tmp]$ $OMNIHOME/install/nco_patch -install
omnibus-3.x-linux2x86-probe-nco-p-syntax-3_0.tar.Z
Installing Patch "omnibus-3.x-linux2x86-probe-nco-p-syntax-3_0" ...
  Short Description : Netcool Syntax probe
  Revision : 0
  Requires : <None>
  Obsoletes : probe-nco-p-syntax probe-nco-p-syntax-2
  Installation Date : Wed May 20 20:32:59 CEST 2009
  Creation Info : Thu Feb 28 09:59:21 GMT 2008
----- README -----
PATCH omnibus-3.x-linux2x86-probe-nco-p-syntax-3_0 - Netcool Syntax
probe

~~~~~

----- End of README -----
Are you sure you want to install this patch? (y/n)? [default: y]
Patch "probe-nco-p-syntax-3" is successfully installed
```

After installation, you can use **nco_p_syntax** command to check the syntax of your custom rules file, as shown in Example 4-25.

Example 4-25 Syntax check

```
[netcool@omnibus probes]$ ./nco_p_syntax -rulesfile
/opt/netcool/omnibus/probes/linux2x86/simnet.rules -server NCOMS
05/29/2009 12:50:46 AM: Information: Connecting ...
05/29/2009 12:50:46 AM: Information: Checking rules file ...
05/29/2009 12:50:46 AM: Debug: Reading
/opt/netcool/omnibus/probes/linux2x86/simnet.rules
05/29/2009 12:50:46 AM: Debug: Plain text rules file detected.
05/29/2009 12:50:46 AM: Debug: Lookup table from
'/opt/netcool/omnibus/probes/linux2x86/cert' has 1 column
05/29/2009 12:50:46 AM: Debug: Including
/opt/netcool/omnibus/probes/linux2x86/rome.rules
05/29/2009 12:50:46 AM: Debug: Plain text rules file detected.
05/29/2009 12:50:46 AM: Debug: End of
/opt/netcool/omnibus/probes/linux2x86/rome.rules
05/29/2009 12:50:46 AM: Debug: End of
/opt/netcool/omnibus/probes/linux2x86/simnet.rules
```

```
05/29/2009 12:50:46 AM: Debug: Number of currently connected servers in
list is 0
05/29/2009 12:50:46 AM: Information: Using targets specified by
properties
05/29/2009 12:50:46 AM: Debug: Creating target for server NCOMS.
05/29/2009 12:50:46 AM: Debug: Setting default target server to
'NCOMS'.
05/29/2009 12:50:46 AM: Debug: Default target backup server is ''.
05/29/2009 12:50:46 AM: Debug: Primary server is 'NCOMS' backup is ''.
05/29/2009 12:50:46 AM: Debug: Attempting a connection to server
'NCOMS'.
05/29/2009 12:50:46 AM: Debug: Checking for backup ObjectServer.
05/29/2009 12:50:46 AM: Information: 'NCOMS' is a primary server.
Polling disabled.
05/29/2009 12:50:46 AM: Debug: Server Verification Starting.
05/29/2009 12:50:46 AM: Debug: Server Verification Complete.
05/29/2009 12:50:46 AM: Debug: Checking for svc update support.
05/29/2009 12:50:46 AM: Debug: Server SUPPORTS services.
05/29/2009 12:50:46 AM: Debug: svc update SUPPORTED
05/29/2009 12:50:46 AM: Debug: SAF: Forwarding SAF file on Initial
startup
05/29/2009 12:50:46 AM: Debug: SAF: Deathtime = 0 : Expire time = 0
05/29/2009 12:50:46 AM: Debug: SAF: Forwarding events from SAF files
05/29/2009 12:50:46 AM: Debug: Heartbeat mode is: standard
05/29/2009 12:50:46 AM: Debug: Heartbeat mode is standard, probe will
function as normal without heartbeating
05/29/2009 12:50:46 AM: Debug: Final number of connected servers in
list is 1
05/29/2009 12:50:46 AM: Information: Rules file syntax OK
05/29/2009 12:50:46 AM: Information: Disconnecting ...
05/29/2009 12:50:46 AM: Debug: Flushing events to object server
```

4.5 ObjectServer to ObjectServer communication

You can connect an ObjectServer to other ObjectServer using a gateway. You can use one of two types of gateways:

- ▶ A bi-directional gateway (4.5.1, “Bi-directional gateway” on page 155) allows resynchronization (if it is enabled by the Gate.Resync.Enable property), which repopulates the failover ObjectServer on old events that are not updated while they still exist in the primary ObjectServer. This is defined for a failover configuration.

- ▶ A uni-directional gateway (4.5.2, “Uni-directional gateway” on page 157) is used to synchronize an ObjectServer with the desktop ObjectServer.

4.5.1 Bi-directional gateway

The bi-directional gateway connects a primary ObjectServer (NCOMS_P) to a backup ObjectServer (NCOMS_B). This section discusses the necessary configuration to create a high availability architecture so that automated failover and a failback architecture is configured. The steps are:

1. On the primary ObjectServer host, create the primary ObjectServer using the following command:

```
nco_dbinit -server NCOMS_P
```
2. On the backup ObjectServer host, create the backup ObjectServer using the following command:

```
nco_dbinit -server NCOMS_B
```
3. Edit `$NCHOME/omnibus/etc/NCOMS_B.props` and set the Backup ObjectServer property to True.
4. Configure a bi-directional gateway between the primary and backup ObjectServer, as shown in 4.1.2, “Gateway configuration” on page 91.
5. On the primary ObjectServer host, configure `$NCHOME/etc/omni.dat` for the primary and backup ObjectServer entries, as shown in Example 4-26.

Example 4-26 The omni.dat file for two ObjectServers

```
[NCOMS_P]
{
    Primary: 9.42.170.132 4501
}
[NCOMS_B]
{
    Primary: 9.42.170.132 4502
}
```

6. On the primary ObjectServer host, regenerate the interface file by running `nco_igen`. Copy the `$NCHOME/etc/interfaces.<arch>` file to the backup ObjectServer host.

- On the backup ObjectServer, enable the backup_startup, backup_counterpart_down, and backup_counterpart_up triggers for automation failover and failback, as shown in Figure 4-50.

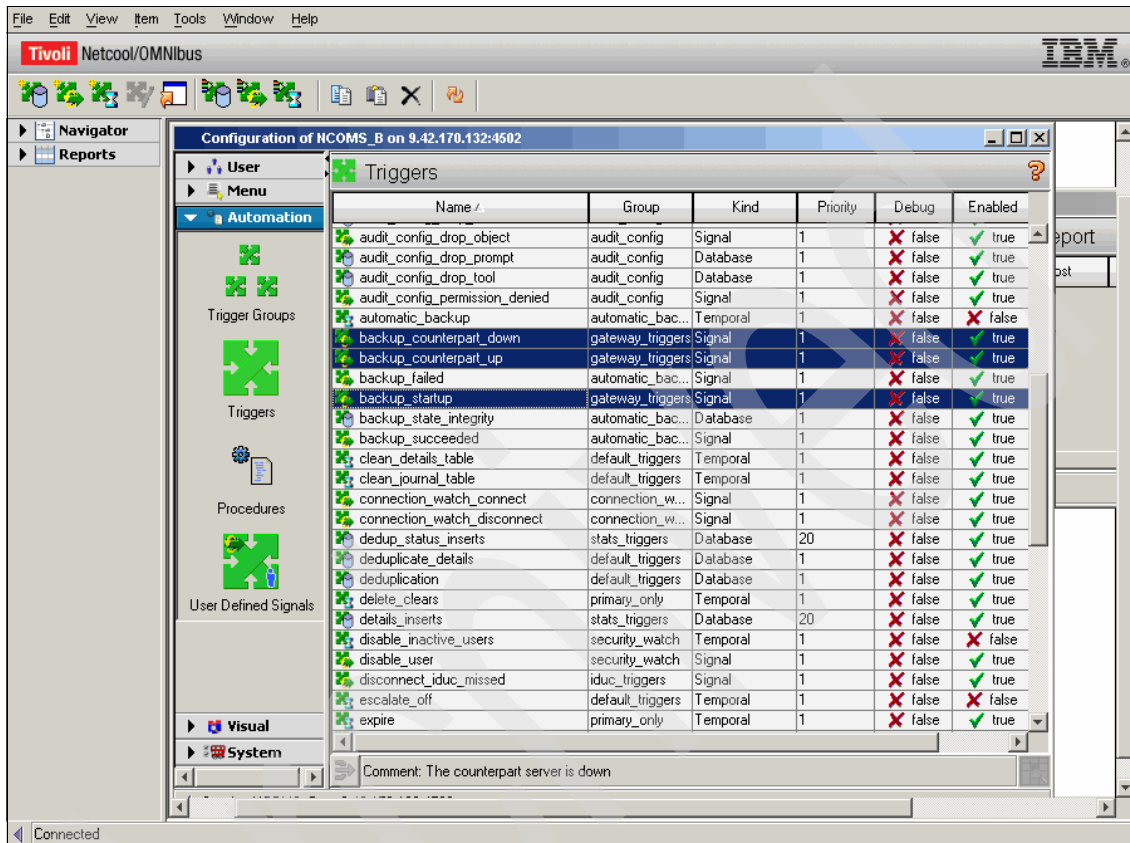


Figure 4-50 Backup trigger enabled

- On the backup ObjectServer, disable the primary_only trigger group for automation failover and failback, as shown in Figure 4-51 on page 157.

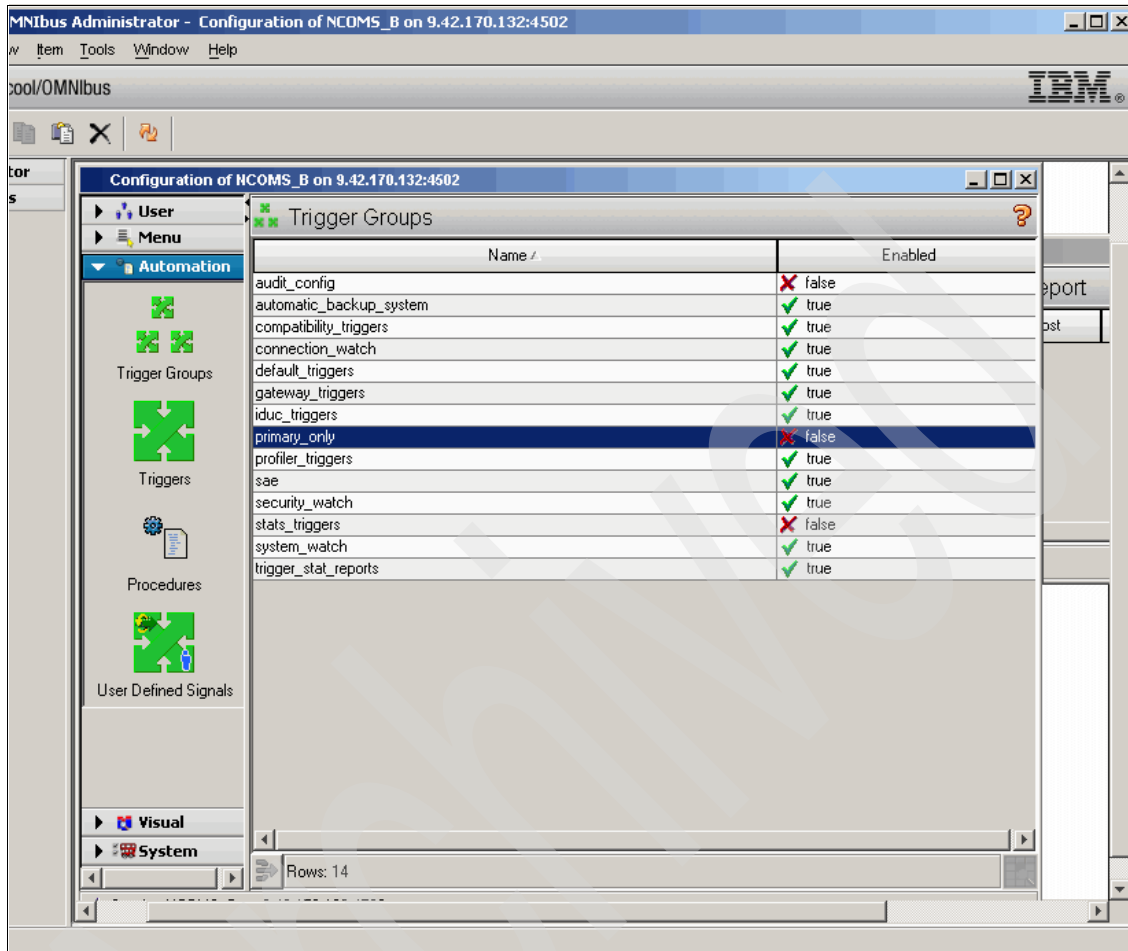


Figure 4-51 Primary trigger group disabled on backup server

4.5.2 Uni-directional gateway

To configure a uni-directional gateway between the primary and desktop ObjectServers to set up a desktop architecture, perform the following steps:

1. Create and configure the primary ObjectServer using the following command:

```
nco_dbinit -server NCOMS_P
```

2. Create the desktop ObjectServer using the following command:

```
nco_dbinit -desktopserver -server DESKOS -dsdprimary NCOMS_P  
-dsddualwrite
```

When a desktop ObjectServer is created with the `-desktopserver` option, the initialization defines a new table called `master.national` and a column in the `alerts.status` called `MasterSerial`. The `MasterSerial` column must be the last column in the `alerts.status` table. The `master.national` contains the following columns:

- KeyField
- MasterServer
- DualWrite

The `master.national` `MasterServer` refers to the master ObjectServer and `DualWrite` indicates whether or not an action item is immediately written to both ObjectServers or not.

3. Create the interface file definition from the `omni.dat` source, as shown in Example 4-27.

Example 4-27 omni.dat

```
[NCOMS_P]
{
    Primary: 9.42.170.132 4501
}
[DESKOS]
{
    Primary: 9.42.170.132 4503
}
```

4. Generate the interface file using the command `nco_igen`. Copy the `$NCHOME/etc/interfaces.<arch>` file from the primary ObjectServer host to the desktop ObjectServer host.
5. Start the desktop ObjectServer, as shown in Example 4-28.

Example 4-28 Starting the desktop ObjectServer

```
[netcool@omnibus etc]$ nco_objserv -name DESKOS
Netcool/OMNIBus Object Server - Version 7.2
(C) Copyright IBM Corp. 1994, 2007

Server 'DESKOS' initialised - entering RUN state.
```

6. Configure a uni-directional gateway between the primary ObjectServer and the desktop ObjectServer, as shown in 4.1.2, “Gateway configuration” on page 91.

7. The event list desktop connects to the desktop ObjectServer and, upon finding the master.national table, it enters Dual Desktop Mode. Dual Desktop Mode allows the event list to connect to the desktop ObjectServer and the primary ObjectServer simultaneously.

4.6 Accelerated Event Notification client

The Accelerated Event Notification (AEN) client provides a means of accelerating high-priority events to help ensure that systems can continue to run without interruption. To configure the Accelerated Event Notification client, system administrators define conditions that identify key events for acceleration, and use channels to propagate these events to specific recipients for action. A channel defines which types or columns of event data to broadcast for accelerated events, and the recipients of this data.

Recipients of the accelerated event data manage the events from the Accelerated Event Notification client. All events that are identified for acceleration are directly sent to the client as notifications. These notifications contain event data that maps to the columns defined for the channel that is currently in use. Recipients can additionally configure settings for the notifications, and can access the Tivoli Netcool/OMNIBus event list or the Netcool/Webtop Active event list to obtain full details for the event and manage the event. This section discusses AEN Client configuration. Perform the following steps:

1. The AEN Client is started by using the command `nco_aen` under `$NCHOME/omnibus/bin` or by selecting **Programs** → **Netcool Suite** → **Notifier** in a GUI. In UNIX, this provides a floating icon, while in Windows it creates an icon in the System Tray. The UNIX floating icon is shown in Figure 4-52.



Figure 4-52 AEN icon

2. Before starting the AEN client for the first time, right-click the AEN notifier icon and select **Properties**. The Application tab of the properties page is shown in Figure 4-53.

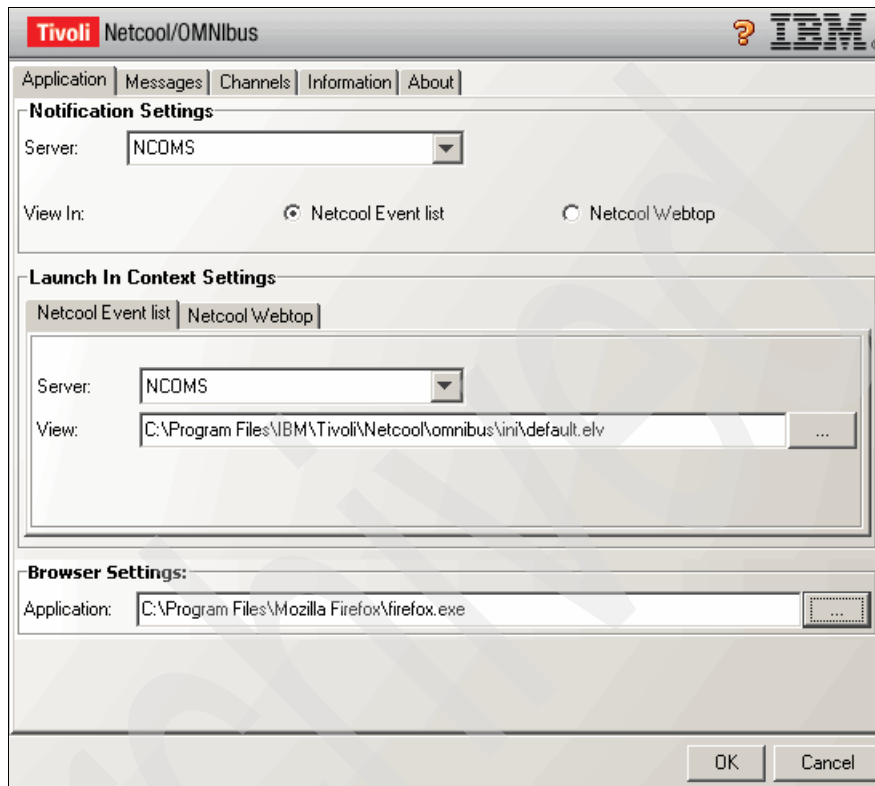


Figure 4-53 AEN client property: Application tab

3. On the Application tab, make the following selections:
 - The server from the Server drop-down list. The accessible servers are defined in the interfaces file. You can view the settings in either Netcool Event List or Netcool Webtop
 - In the Browser Settings, select the view to open for the ObjectServer and provide the Web browser executable.

4. On the Messages tab, define the History Settings, as shown in Figure 4-54. The AEN Client is not configured to maintain the details of expired messages.

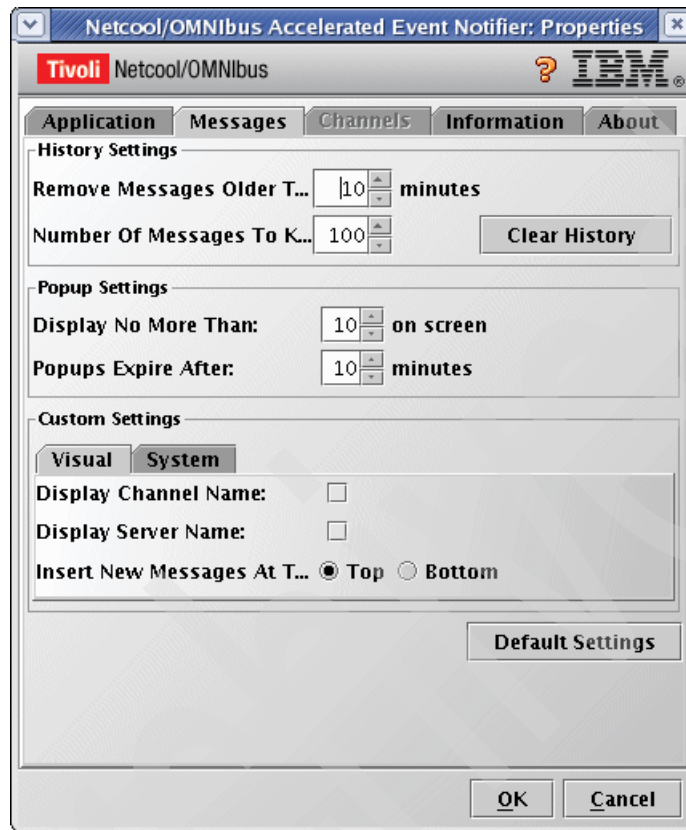


Figure 4-54 Messages tab

5. On the Channels tabs, modify the settings as shown in Figure 4-55.

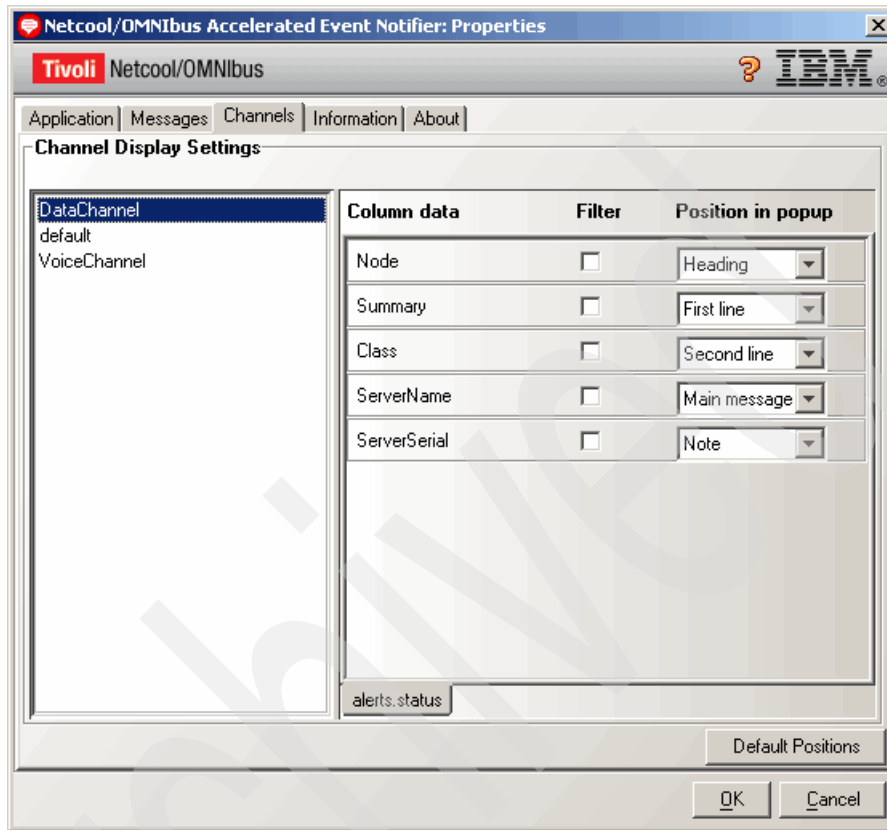


Figure 4-55 AEN Channels tab

6. Select **OK**.
7. After the setting has been saved, you can start the client by right-clicking the icon and selecting **Sign In**. Provide the appropriate credential for the sign in window shown in Figure 4-56 on page 163 and click **Log In**.

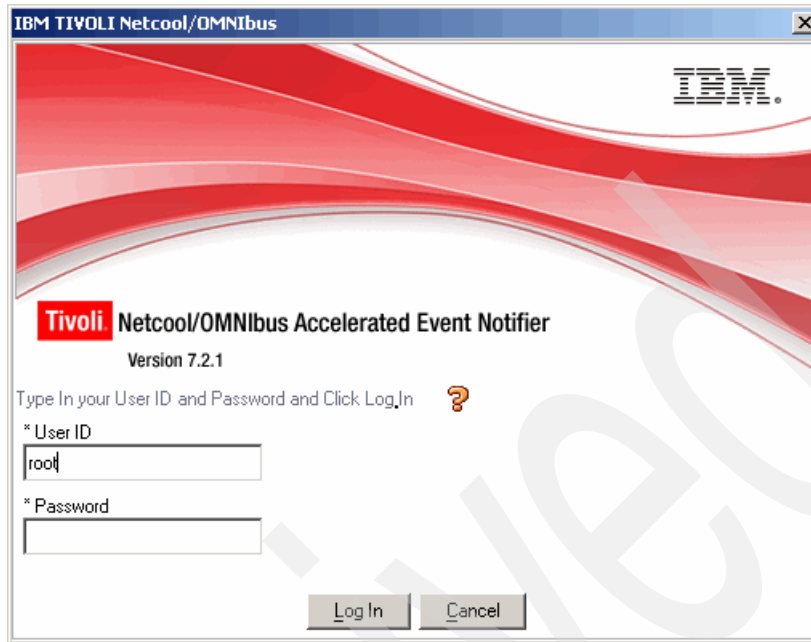


Figure 4-56 AEN login window

8. Note that the AEN icon changes from a yellow triangle to a green box.
9. AEN events are defined from the probe rules that insert a flag for notification. This flag is then used by a database trigger to be sent to a channel.
 - First, modify the probe rule to add a flag field. We use a Channel field that varies depending on the content of the Summary field. The probe rules are shown in Example 4-29.

Example 4-29 Probe rule

```
if (regmatch($Summary, "^Port Failure.*"))
{
    @Channel = 1
}
if (regmatch($Summary, "^Disk.*"))
{
    @Channel = 2
}
```

- Define a database trigger to respond to either the Flag field or the appropriate database action or condition, using the SQL commands IDUC EVTFT or IDUC SNDMSG to generate AEN, as shown in Figure 4-57.

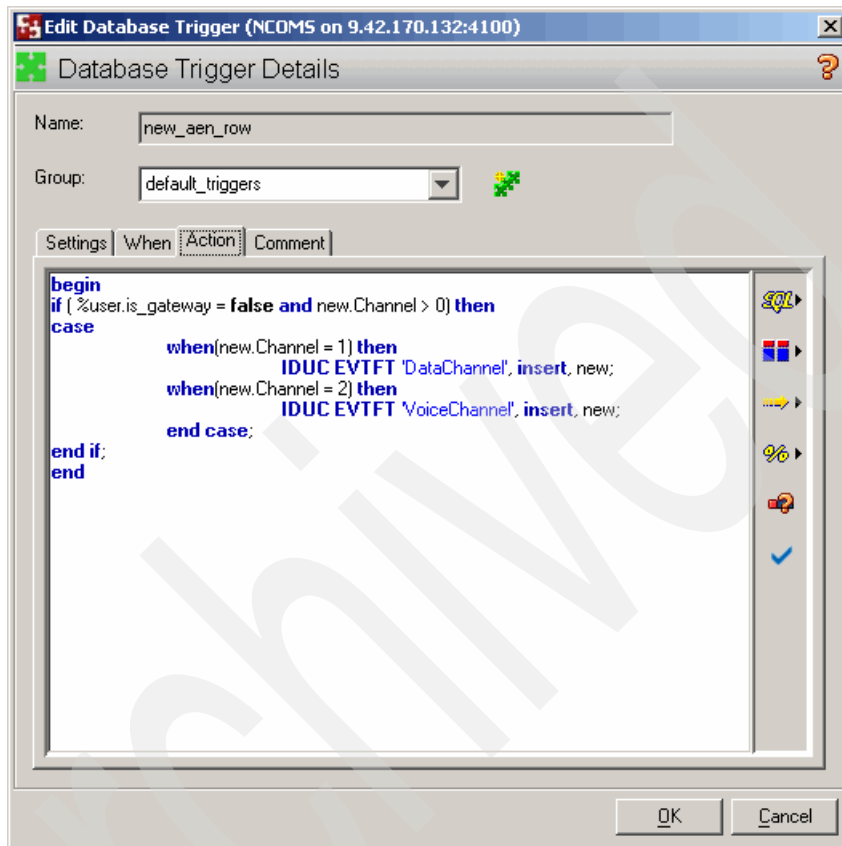


Figure 4-57 Create trigger action

10. Configure a Channel to broadcast AEN event data by performing the following steps:
 - a. From the IBM Tivoli Netcool/OMNIbus Administrator window, select **System** → **Channels**, as shown in Figure 4-58 on page 165.

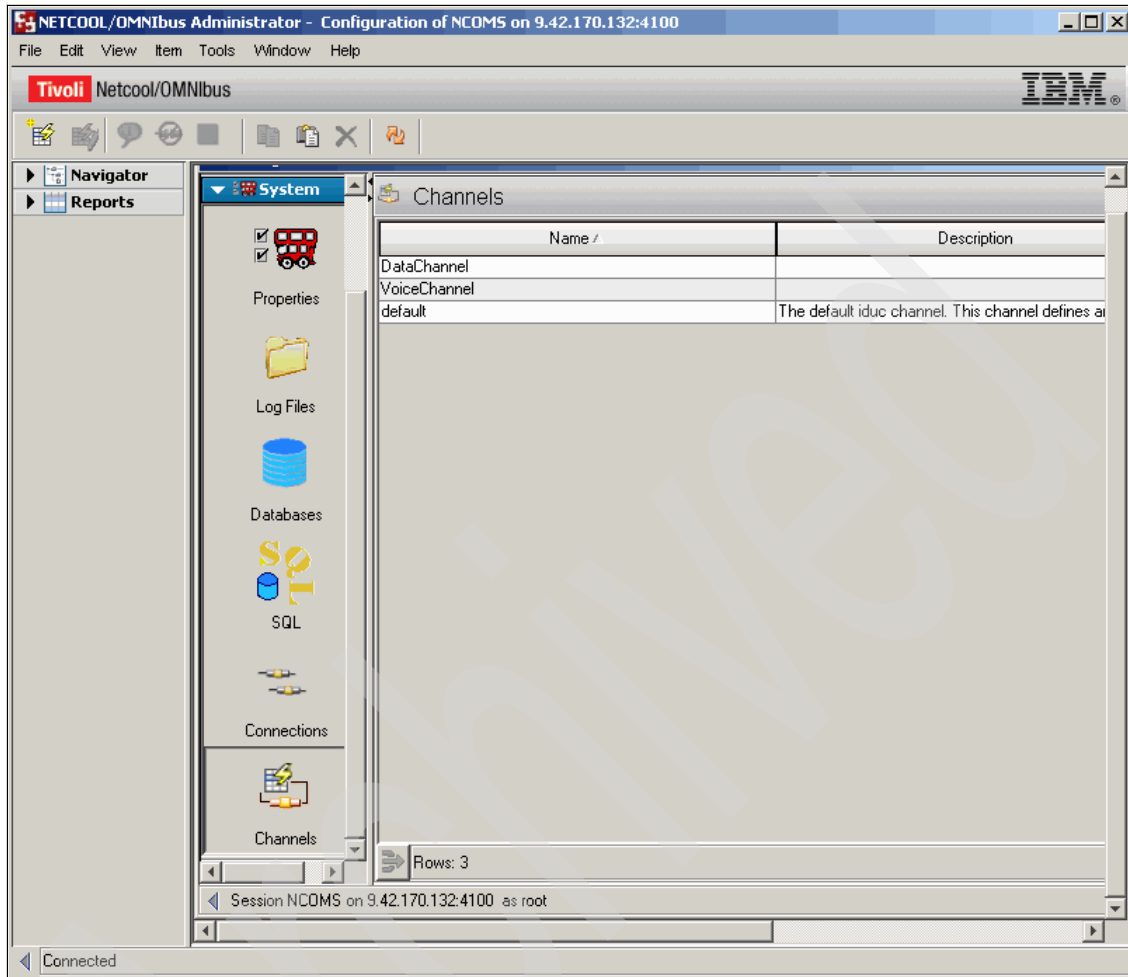


Figure 4-58 Administration console Channels tab

- b. Right-click within the GUI to select **Add a new Channel**.

- c. Within the Channel Details window, add a name for the channel, as shown in Figure 4-59.

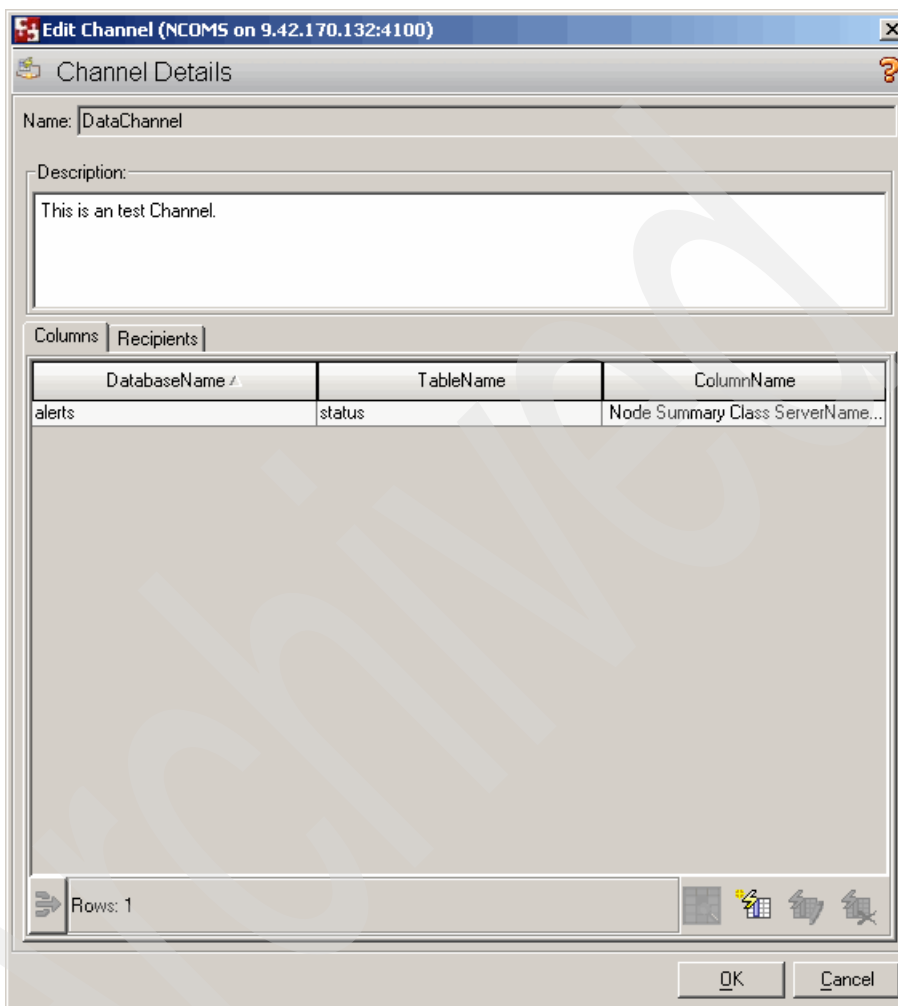



Figure 4-59 Channel Details tab

- d. On the Columns tab, select the  button to be able to define the ObjectServer Table and Fields to use for the notification, as shown in Figure 4-60. Click **OK**.

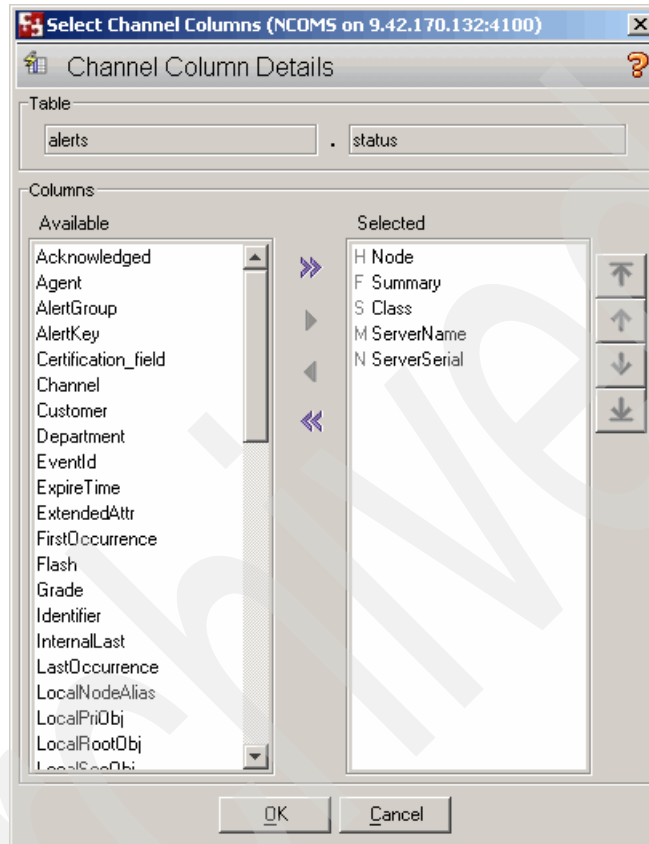


Figure 4-60 Channel column details

- e. On the Recipients tab, select the  button to add new recipient, as shown in Figure 4-61.

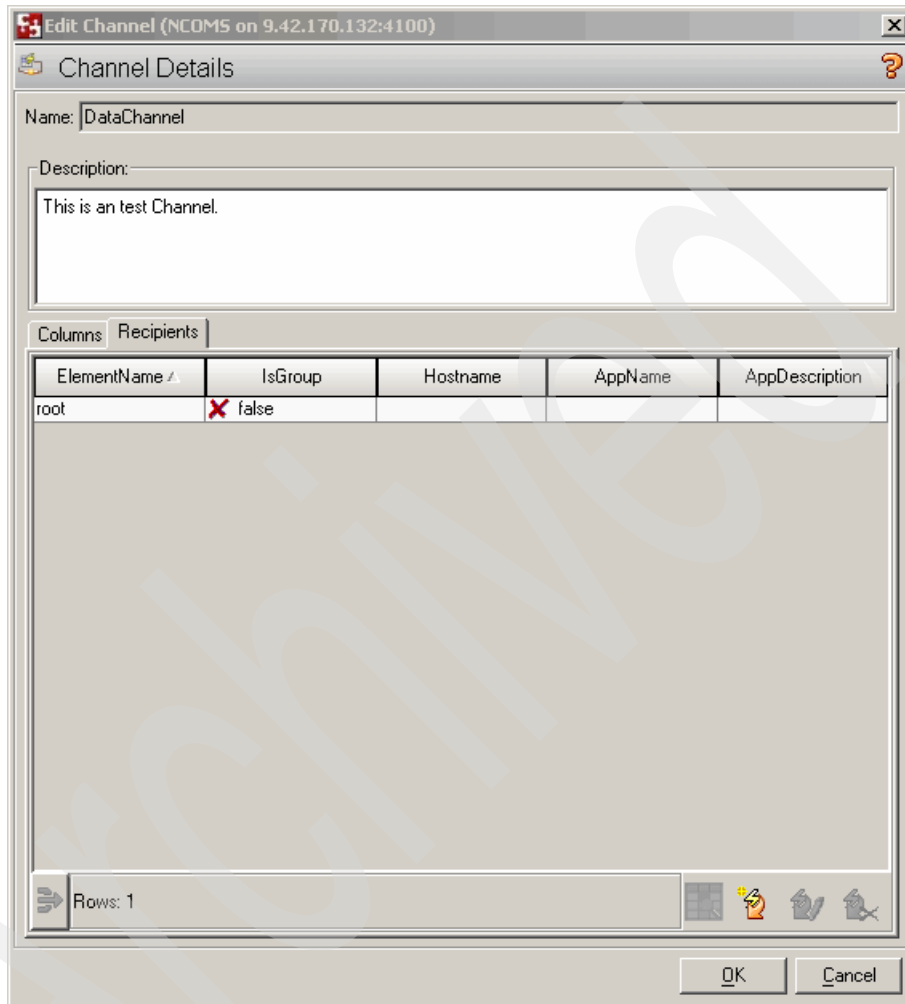


Figure 4-61 Recipient created

- f. The New Channel Recipients window opens, as shown in Figure 4-62 on page 169. Define the recipients for the notification in this window and click **OK**.

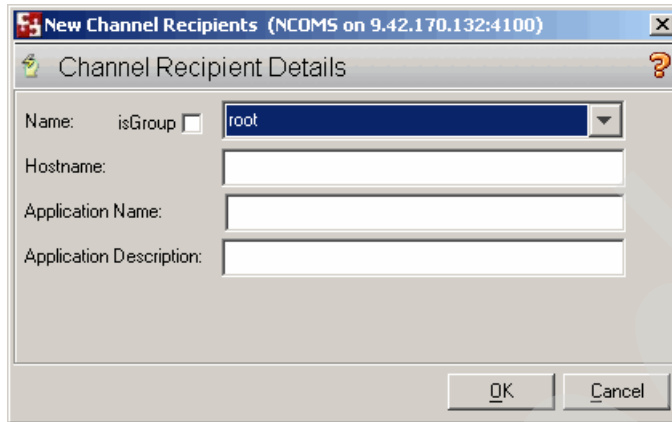


Figure 4-62 Channel recipient window

- g. When complete, click the **OK** button on the Channel Details window shown in Figure 4-59 on page 166.

11. Test the Channel notification by right-clicking the channel and selecting **Send Message**, as shown in Figure 4-63.

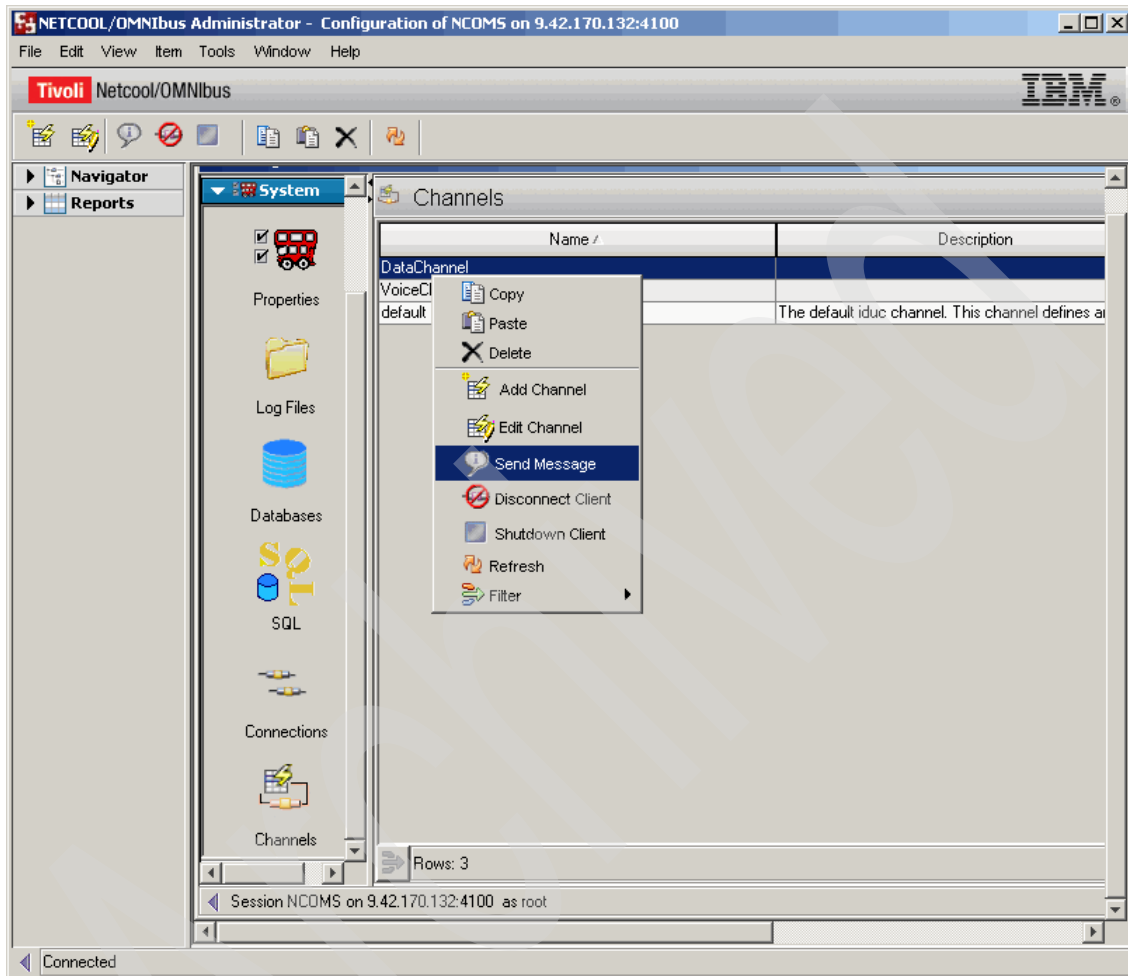


Figure 4-63 Test channel

12. Insert a test message and click **OK** to send it. You should get a notification event, as shown in Figure 4-64 on page 171.

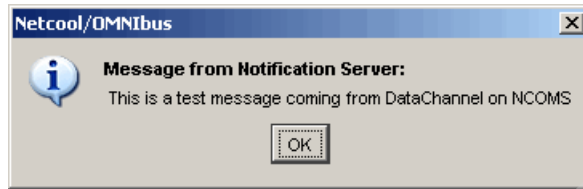


Figure 4-64 Test message

4.7 IBM Tivoli Health Monitoring agent for ObjectServer V7.2

Tivoli Health Monitoring Agent for ObjectServer V7.2 can be used to monitor IBM Tivoli Netcool/OMNibus ObjectServer. The agent is based on IBM Tivoli Monitoring V6.1. The agent code is called NO. To enable this monitoring, you must import the schema and automation into the ObjectServer. Use the commands `nco_sql` in UNIX or `isql` in Windows. The source of the update is `itm_os.sql`, which resides in `$ITMHOME/<arch>/no/bin/` or `%ITMHOME%\TMAITM6\`.

Example 4-30 shows importing the schema for UNIX.

Example 4-30 importing schema

```
[netcool@omnibus db]$ /opt/netcool/omnibus/bin/nco_sql -user root
-server NCOMS_P < /opt/IBM/ITM/li6263/no/bin/itm_os.sql
Password:
(0 rows affected)
(0 rows affected)
(0 rows affected)
(0 rows affected)
(0 rows affected)
(0 rows affected)
```

Figure 4-65 shows the Health Monitoring agent on Tivoli Enterprise Portal.

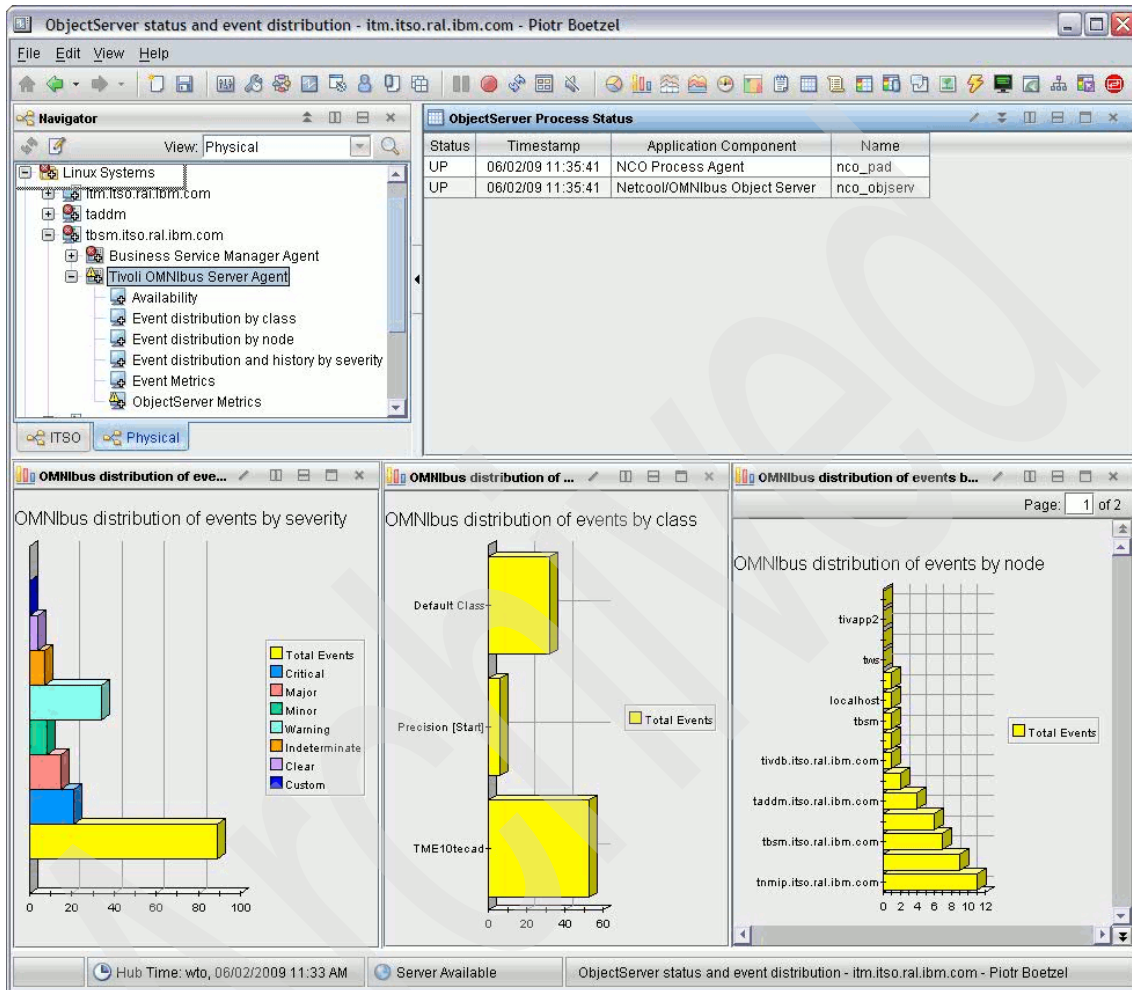


Figure 4-65 Health Monitoring agent

Operation

This chapter discusses the operation, administration, and problem determination of IBM Tivoli Netcool/OMNIbus. The topics discussed are:

- ▶ 5.1, “Device definition in simnet” on page 174
- ▶ 5.2, “Failover operation” on page 176
- ▶ 5.3, “Problem determination” on page 178
- ▶ 5.4, “Performance tuning” on page 184

5.1 Device definition in simnet

In order to generate a test event and send it to an ObjectServer, you need to configure the simnet probe and add devices to the `simnet.def` file.

Perform the following steps:

1. Edit the `$NCHOME/omnibus/probes/ARCH/simnet.def` file and define the devices. A sample `simnet.def` file for two devices (`device1` and `device2`) is shown in Example 5-1.

Example 5-1 simnet.def

```
[netcool@omnibus db]$ cd /opt/netcool/omnibus/probes/linux2x86/  
[netcool@omnibus linux2x86]$ cat simnet.def  
device1 0 50  
device2 3 100
```

2. Modify `$NCHOME/omnibus/probes/ARCH/simnet.props` to set the LogFile and Server properties, as shown in Example 5-2.

Example 5-2 simnet.props

```
[netcool@omnibus linux2x86]$ vi simnet.props  
LogFile      : '$OMNIHOME/probes/linux2x86/simnet.def'  
Server       : 'NCOMS'
```

3. Start the simnet probe using the command `nco_p_simnet` under `$NCHOME/omnibus/probes/`.
4. Check that events are being sent correctly to the object server, as shown in Figure 5-1 on page 175.

Netcool/OMNIBus Event List : Filter="Default", View="simnet"

File Edit View Alerts Tools Help

Default simnet Top

Manager	Node	Last Occurrence	Summary
Simnet Probe:device2	device2	30/05/2009 11.32.00	Port failure : port reset Probe running on :NCDMS
Simnet Probe:link5	link5	30/05/2009 11.34.34	Link Up on port Probe running on :NCDMS
Simnet Probe:link3	link3	30/05/2009 11.34.55	Link Up on port Probe running on :NCDMS
Simnet Probe:link1	link1	30/05/2009 11.34.29	Link Up on port Probe running on :NCDMS
Simnet Probe:Tokyo	Tokyo	30/05/2009 11.35.27	Diskspace alert Probe running on :NCDMS
Simnet Probe:Washington	Washington	30/05/2009 11.34.01	Machine has gone online Probe running on :NCDMS
Simnet Probe:link4	link4	30/05/2009 11.35.28	Link Down on port Probe running on :NCDMS
Simnet Probe:London	London	30/05/2009 11.35.10	Switch rule test for node: London
Simnet Probe:link3	link3	30/05/2009 11.34.42	Link Down on port Probe running on :NCDMS
Simnet Probe:link4	link4	30/05/2009 11.35.25	Link Up on port Probe running on :NCDMS
Simnet Probe:Moscow	Moscow	30/05/2009 11.34.18	Switch rule test for node: Moscow
Simnet Probe:Sydney	Sydney	30/05/2009 11.34.33	Machine has gone online Probe running on :NCDMS
Simnet Probe:Beijing	Beijing	30/05/2009 11.35.29	Diskspace alert Probe running on :NCDMS
Simnet Probe:Tokyo	Tokyo	30/05/2009 11.35.24	Diskspace alert Probe running on :NCDMS
Simnet Probe:London	London	30/05/2009 11.34.41	Switch rule test for node: London
Simnet Probe:Sydney	Sydney	30/05/2009 11.34.19	Machine has gone offline Probe running on :NCDMS
Simnet Probe:Washington	Washington	30/05/2009 11.35.09	Machine has gone offline Probe running on :NCDMS
Simnet Probe:Moscow	Moscow	30/05/2009 11.34.45	Switch rule test for node: Moscow
Simnet Probe:link1	link1	30/05/2009 11.35.22	Link Down on port Probe running on :NCDMS

7 0 7 3 10 0

0 rows selected 02/06/2009 15.45.53 root NCDMS [PRI]

Figure 5-1 Simnet events

5.2 Failover operation

Given a failover architecture composed of an ObjectServer Failover Pair, a bi-directional gateway, and probes, verify that the failover architecture is working by performing these steps:

1. Connect to the primary server and check that events are coming in, as shown in Figure 5-2.

Serial	Node	Count	Last Occurrence	
35	Berlin	2	30/05/2009 11.55.44	Port failure : port rese
36	link3	1	30/05/2009 11.55.38	Link Up on port P
37	link4	2	30/05/2009 11.55.43	Link Up on port P
38	link2	1	30/05/2009 11.55.40	Link Up on port P
39	link1	1	30/05/2009 11.55.41	Link Up on port P
40	link4	2	30/05/2009 11.55.58	Link Down on port f
41	Sydney	1	30/05/2009 11.55.45	Machine has gone onli
42	link5	1	30/05/2009 11.55.46	Link Up on port P
43	Beijing	5	30/05/2009 11.55.55	Diskspace alert Pr
44	Tokyo	3	30/05/2009 11.55.56	Diskspace alert Pr
45	London	1	30/05/2009 11.55.53	Switch rule

Summary: 5 (green), 0 (purple), 4 (blue), 2 (yellow), 2 (orange), 0 (red)

13 rows matched | 02/06/2009 16.20.26 | root | NCOMS_P [PRI]

Figure 5-2 NCOMS_P events

2. Shut down the Master ObjectServer. The message window shown in Figure 5-3 opens.



Figure 5-3 Disconnect from ObjectServer

3. Wait until the desktop client connects automatically to the backup ObjectServer, as shown in Figure 5-4 on page 177.

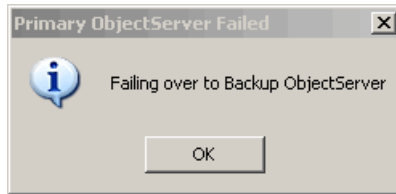


Figure 5-4 Failover

4. Check that the events are coming into the backup ObjectServer, as shown in Figure 5-5.

Serial	Node	Count	Last Occurrence	Summary
3	IBM-5E6948F4416	1	29/05/2009 21.16.27	A Administrator process running on IBM-5E6948F4416
4	IBM-5E6948F4416	1	29/05/2009 21.42.15	A Administrator process running on IBM-5E6948F4416
5	omnibus	1	30/05/2009 11.57.22	ObjectServer NCOMS_B on omnibus started at Sat May
6	omnibus	1	30/05/2009 11.57.22	ObjectServer NCOMS_B Profiler disabled at Sat May
7	IBM-5E6948F4416	1	30/05/2009 11.57.51	A NT Event List@092CA843 process running on IBM-5
8	omnibus	1	30/05/2009 11.57.56	A PROBE process simnet running on omnibus has come
9	Tokyo	22	30/05/2009 11.58.40	Diskspace alert Probe running on :NCO
10	Beijing	33	30/05/2009 11.58.39	Diskspace alert Probe running on :NCO
11	link4	10	30/05/2009 11.58.26	Link Up on port Probe running on :NCO
12	Rome	3	30/05/2009 11.57.38	Summary created by /opt/netcool/omnibus/probes/li
13	Berlin	5	30/05/2009 11.58.18	Port failure : port reset Probe running on :N
14	Washington	2	30/05/2009 11.57.22	Machine has gone offline Probe running on :
15	London	1	30/05/2009 11.56.22	Switch rule test for node: London
16	link5	1	30/05/2009 11.56.28	Link Down on port Probe running on :NCO
17	link4	10	30/05/2009 11.58.42	Link Down on port Probe running on :NCO
18	link1	4	30/05/2009 11.58.43	Link Down on port Probe running on :NCO
19	Tokyo	21	30/05/2009 11.58.09	Diskspace alert Probe running on :NCO
20	Moscow	1	30/05/2009 11.56.43	Switch rule test for node: Moscow
21	Sydney	1	30/05/2009 11.56.48	Machine has gone offline Probe running on :
22	link5	1	30/05/2009 11.56.49	Link Up on port Probe running on :NCO
23	link1	3	30/05/2009 11.58.37	Link Up on port Probe running on :NCO
24	Beijing	8	30/05/2009 11.58.05	Diskspace alert Probe running on :NCO
25	link3	3	30/05/2009 11.58.04	Link Down on port Probe running on :NCO
26	link6	4	30/05/2009 11.58.34	Link Up on port Probe running on :NCO
27	Washington	2	30/05/2009 11.57.54	Machine has gone online Probe running on :
28	Moscow	1	30/05/2009 11.57.16	Switch rule test for node: Moscow
29	London	1	30/05/2009 11.57.26	Switch rule test for node: London

Figure 5-5 Backup ObjectServer event

5. Restart the Master ObjectServer.

6. Wait until the desktop client connects to the Master ObjectServer, as shown in Figure 5-6.

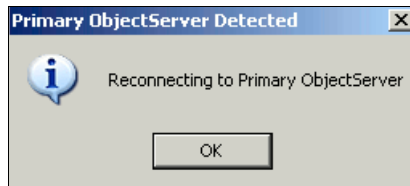


Figure 5-6 Reconnect or fallback message

7. Check that the client indicates that is connected to the Master ObjectServer.

5.3 Problem determination

This section discusses some problem determination issues for the following issues:

- ▶ 5.3.1, “Startup problems” on page 178
- ▶ 5.3.2, “Probe connection” on page 180
- ▶ 5.3.3, “Desktop startup” on page 181
- ▶ 5.3.4, “Slow response time” on page 181

5.3.1 Startup problems

On a UNIX/Linux system, the ObjectServer V7.2 is installed, but will not start. Determine why the ObjectServer does not start by performing the following steps:

- ▶ Check the environment variables `$NCHOME`, as shown in Example 5-3.

Example 5-3 Checking \$NCHOME

```
[netcool@omnibus etc]$ set | grep NC  
NCHOME=/opt/netcool/
```

- ▶ Check to make sure that the ObjectServer has been created by running `$NCHOME/omnibus/bin/nco_dbinit`, as shown in Example 5-4 on page 179.

Example 5-4 Checking ObjectServer database

```
[netcool@omnibus etc]$ cd /opt/netcool/omnibus/db/
[netcool@omnibus db]$ ls -l NCOMS
total 19072
-rw-r----- 1 netcool root 2887680 May 30 18:05 master_store_0.chk
-rw-r----- 1 netcool root   1332 May 30 18:01 master_store_0.log
-rw-r----- 1 netcool root 2887680 May 30 18:01 master_store_1.chk
-rw-r----- 1 netcool root   1124 May 30 18:05 master_store_1.log
-rw-r----- 1 netcool root 2822144 May 28 22:02 master_store.tab
-rw-r----- 1 netcool root 3674112 May 30 18:07 table_store_0.chk
-rw-r----- 1 netcool root    428 May 30 18:08 table_store_0.log
-rw-r----- 1 netcool root 3674112 May 30 18:06 table_store_1.chk
-rw-r----- 1 netcool root    740 May 30 18:07 table_store_1.log
-rw-r----- 1 netcool root 3543040 May 28 22:02 table_store.tab
```

- ▶ Check to make sure that the entry for the ObjectServer exists in `$NCHOME/etc/omni.dat`, as shown in Example 5-5. Ensure that the interface file has been generated by running the `nco_igen` command.

Example 5-5 Checking the interface file

```
[netcool@omnibus db]$ cd /opt/netcool/etc/
[netcool@omnibus etc]$ more omni.dat
[NCOMS]
{
    Primary: 9.42.170.132 4100
}
```

- ▶ Ensure that the port for the ObjectServer is not already in use by running the `netstat` command, as shown in Example 5-6.

Example 5-6 Checking port usage

```
[netcool@omnibus etc]$ netstat -na | grep 4100
```

- ▶ Ensure that the proper user is starting the object server process. You can check if this is the case by running the `ps -ef` command and checking for the presence of the `nco_objserv` process.

5.3.2 Probe connection

If a specific probe does not start up correctly, it is usually due to some very specific issues that can be investigated by performing the following steps:

- ▶ Check the probes log file. It is usually found under `/opt/netcool/omnibus/log`. Look for any error messages in there. Example 5-7 shows some sample error messages.

Example 5-7 Some probes error messages

```
Error: Failed to read rules - aborting
Error: Couldn't parse line
Error: Connection to object server 'NCOMS_P' marked DEAD
```

- ▶ Verify that the probe's designated ObjectServer is running. Use the `nco_ping` command to verify that the object server is up and running and is reachable from the probe machine.
- ▶ Check whether the interfaces file on the probe server is correctly defined. Look for any errors inside `/opt/netcool/etc/omni.dat`. If any error is found, fix it and run `nco_igen`.
- ▶ Verify that the probe server has network connectivity to the ObjectServer host. Check the basic connectivity by running a standard command such as `ping` or `traceroute`.
- ▶ Check whether any firewall settings are affecting communications. Use `telnet` to verify that the ObjectServer port is open and reachable from the probe. Example 5-8 shows that the connection was refused and Example 5-9 shows that the connection is working.

Example 5-8 Connection refused

```
[netcool@omnibus log]$ telnet 9.42.170.132 4501
Trying 9.42.170.132...
telnet: connect to address 9.42.170.132: Connection refused
telnet: Unable to connect to remote host: Connection refused
```

Example 5-9 Connection working

```
[netcool@omnibus log]$ telnet 9.42.170.132 4501
Trying 9.42.170.132...
Connected to omnibus (9.42.170.132).
Escape character is '^['.
```

5.3.3 Desktop startup

On a UNIX server, run the `nco_event` command to open the desktop client. If it does not start correctly, determine the cause of the issue by performing the following steps:

- ▶ Ensure that server's settings are set correctly. Export the UNIX server's `DISPLAY` variable, as shown in Example 5-10.

Example 5-10 Setting DISPLAY variable

```
[netcool@omnibus log]$ export DISPLAY=9.44.168.67:0.0
[netcool@omnibus log]$ set | grep DISPLAY
DISPLAY=9.44.168.67:0.0
```

- ▶ Ensure that the ObjectServer to which the desktop is connected is running. Run the `nco_ping` command to check the connection.

5.3.4 Slow response time

If users are complaining of slow response times while using a specific ObjectServer, determine the reason for the slow event list response by performing the following steps:

- ▶ Check the number of events by running `nco_sql`, as shown in Example 5-11.

Example 5-11 Checking number of events

```
[netcool@omnibus log]$ nco_sql -server NCOMS_P -user root
Password:
1> select count(*) from alerts.status;
2> go
COUNT( * )
-----
                30

(1 row affected)
```

- ▶ Check the ObjectServer profile. If profiling is turned on, you will have profile logs in your logs directory and you can find execution time information for the various ObjectServer processes, as shown in Example 5-12. The log shows the component that consumes the most processing power. By differentiating the user profiles running a group of processes, you can easily find which group uses the most resources.

Example 5-12 Profile log

```
Sat May 30 15:26:16 2009: Profiling enabled at Sat May 30 15:26:16 2009
Sat May 30 15:27:11 2009: Individual user profiles:
Sat May 30 15:27:11 2009: 'isql' (uid = 0) time on omnibus: 0.146961s
Sat May 30 15:27:11 2009: Grouped user profiles:
Sat May 30 15:27:11 2009: Execution time for all connections whose application name
is 'isql': 0.146961s
Sat May 30 15:27:11 2009: Total time in the report period (59.098127s): 0.146961s
Sat May 30 15:28:11 2009: Total time in the report period (59.992863s): 0.000000s
Sat May 30 15:29:11 2009: Total time in the report period (60.001173s): 0.000000s
Sat May 30 15:30:11 2009: Total time in the report period (60.000666s): 0.000000s

Sat May 30 17:52:11 2009: Individual user profiles:
Sat May 30 17:52:11 2009: 'PROBE' (uid = 0) time on omnibus: 0.002685s
Sat May 30 17:52:11 2009: Grouped user profiles:
Sat May 30 17:52:11 2009: Execution time for all connections whose application name
is 'PROBE': 0.002685s
Sat May 30 17:52:11 2009: Total time in the report period (60.000878s): 0.002685s
Sat May 30 17:53:11 2009: Individual user profiles:
Sat May 30 17:53:11 2009: 'PROBE' (uid = 0) time on omnibus: 0.001663s
```

- ▶ Check the gateway resynchronization. Synchronization can consume a great deal of processor and I/O capacity, especially in a busy ObjectServer environment.
- ▶ Check event rates. Profiling information about event rates can be found in the NCOMS_omniEvtRate.log under \$NCHOME/omnibus/log directory. This log provides all the information needed to discover what the event rate is, as shown in Example 5-13 on page 183.

Example 5-13 Finding the event rate

```
1090531172728000:NCOMS:tbsm.itso.ral.ibm.comGATEWAY:0:0:4:0:0
1090531172728000:NCOMS:tbsm.itso.ral.ibm.comGATEWAYGateway Reader/Writer:0:0:2:0:0
1090531172728000:NCOMS:tbsm.itso.ral.ibm.comPROBEtivol_i_eif:2:6:0:86:0
SAMPLE_DELIMITER
1090531172828000:NCOMS:tbsm.itso.ral.ibm.comGATEWAY:0:0:4:0:0
1090531172828000:NCOMS:tbsm.itso.ral.ibm.comGATEWAYGateway Reader/Writer:0:0:2:0:0
1090531172828000:NCOMS:tbsm.itso.ral.ibm.comPROBEtivol_i_eif:2:7:0:86:0
SAMPLE_DELIMITER
1090531172928000:NCOMS:tbsm.itso.ral.ibm.comPROBEtivol_i_eif:0:4:0:0:0
SAMPLE_DELIMITER
1090531173028000:NCOMS:tbsm.itso.ral.ibm.comGATEWAY:0:0:4:0:0
```

- ▶ Check the number of desktops. You can easily check the number of desktop connections by using the administrator console and going to the **System** → **Connections** tab, as shown in Figure 5-7.

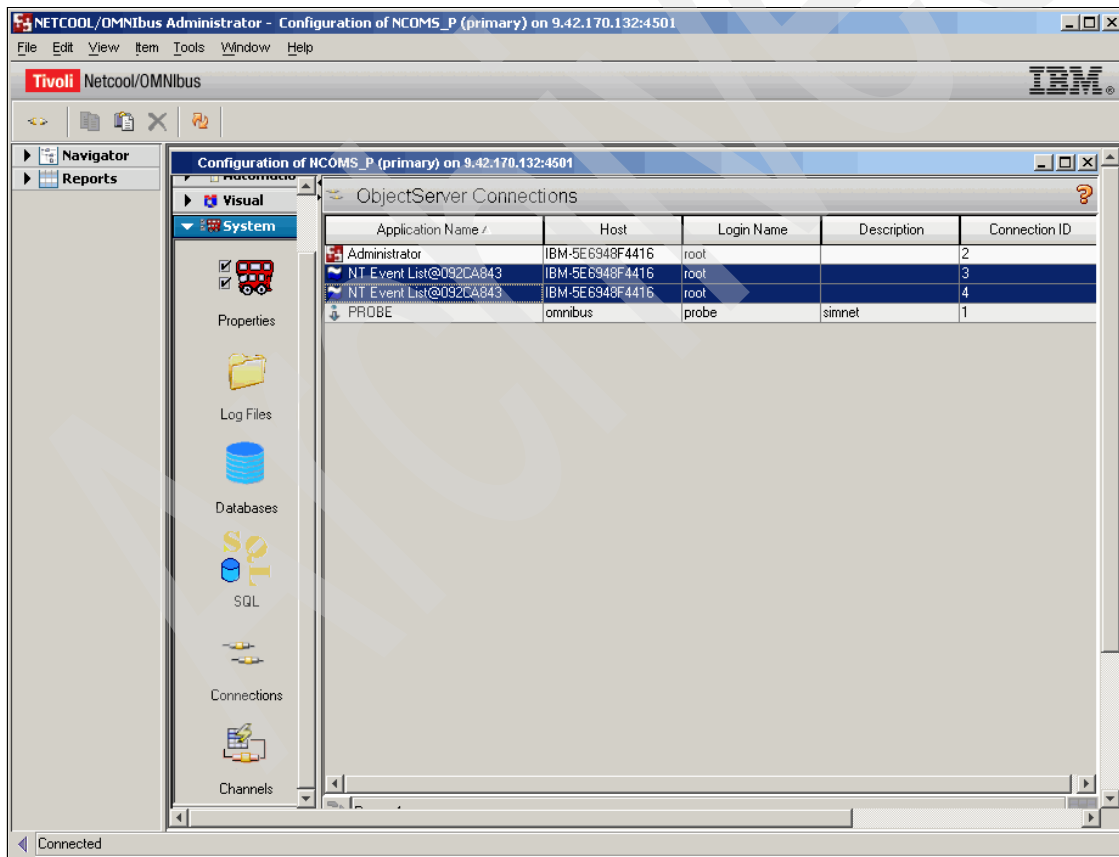


Figure 5-7 Desktop connections

You can also run a simple `nco_sql` query on the `catalog.connections` table, as shown in Example 5-14.

Example 5-14 Checking desktop connections

```
1> select count(*) from catalog.connections where AppName like 'Event
List';
2> go
COUNT( * )
-----
                2
(1 row affected)
```

- ▶ Check the process on the ObjectServer. You can run the **top** or **ps** commands on a UNIX server to discover process information related to the Object Server.
- ▶ Check the resource usage of the ObjectServer, that is, the memory, CPU, and disk access by running commands like **vmstat**, **df**, **du**, or **sar**.
- ▶ Check network response times by running the **ping** and **traceroute** commands.
- ▶ Check automations.
- ▶ Check the number of journals (`alerts.journal`) and details (`alerts.detail`).
- ▶ Determine what other components are connected.

5.4 Performance tuning

In order to discover information about performance on IBM Tivoli Netcool/OMNIbus and optimize your system, perform the following steps:

- ▶ Check the frequency of triggers. Triggers should not be configured to execute too frequently. Tune and spread out temporal trigger execution times so that not too many triggers execute at the same time. This may impact performance considerably.
- ▶ Check the execution scope of triggers: once only, for each row.
- ▶ Check the number of details by viewing the `details($*)` entry in the rules files. Do not use details unless necessary and only for the necessary amount of time.

- ▶ Check the SQL used in triggers, desktop filters, and procedures for performance.
 - The ordering of the SQL clause is very important: Integer fields should be matched first, then the Date field, and then exact string matching followed by expression string matching.
 - Avoid using too many where statements; instead, use the in list statement.
 - Try to build regular expressions to be as specific as possible.
- ▶ Check whether the UPDATE VIA statement is being used.

Archived

Archived

Sample test

This appendix is divided into the following sections:

- ▶ “Sample questions” on page 188
- ▶ “Answer key” on page 191

Sample questions

Here are our sample questions:

1. A customer wants to minimize system administration, including ObjectServer maintenance. The customer is running a highly available pair of ObjectServers. The customer wants the system configured so that any new users that are added to the primary ObjectServer will also be added to the failover server with the least amount of manual intervention and in a timely manner. What is the best technique that can be used to satisfy the customer's requirements?
 - a. Add entries to the gateway `tblrep.def` file and map the file to enable replication of the user tables.
 - b. Run `nco_confpack` to extract the user tables from the primary server and import them into the failover server.
 - c. Create a script to use the `nco_sql` command to connect to the gateway and issue `REPLICATE` commands to copy the user tables
 - d. Configure `TRANSFER` commands in the gateway `startup.cmd` file to copy the user tables from the primary server to the failover server.
2. What are the default authorization roles available to a normal user created in OMNibus?
 - a. AlertsUser, Normal, and Public.
 - b. AlertsUser, CatalogUser, and ISQLWrite.
 - c. AlertsUser, CatalogUser, and ChannelUser.
 - d. AlertsUser, ChannelUser, and DesktopAdmin.
 - e. Every user is assigned the Normal role by default.
3. To set up IBM Tivoli Netcool/OMNibus V7.2 authorization, security objects should be configured in a certain order. In which order should security objects be configured?
 - a. Users, Groups, and Roles.
 - b. Roles, Groups, and Users.
 - c. Groups, Users, and Roles.
 - d. Groups, Roles, and Users.

4. IBM Tivoli Netcool/OMNIBus V7.2 provides several default groups. Which default groups are required by the ObjectServer and cannot be deleted or renamed?
 - a. Administrator, Public, and ISQL.
 - b. ISQL, ISQLWrite, and Public.
 - c. Normal, Administrator, and System.
 - d. Public, AlertsUser, and System.
5. During an event storm, 50 events per second were logged in the system log file. The syslog probe sent 30 events per second to the ObjectServer. What should be changed to improve probe performance?
 - a. The syslog probe property StoreAndForward should be set to 1.
 - b. The syslog probe should be configured to read from a named pipe.
 - c. The syslog probe property Buffering should be enabled and you should adjust the BufferSize property.
 - d. The syslog probe property Mode should be set to Master, and a new syslog probe should be configured with Mode set to slave.
6. The monitor you are using has to connect to an Element Management System (EMS) to retrieve events. The host name of the EMS is "emsserver". The device listens for connections on port 23. The monitor is not receiving any events, and the log file tells you that it cannot log into the EMS. How can you manually check this error?
 - a. Run the `ssh emsserver` command.
 - b. Run the `telnet emsserver` command.
 - c. Run the `nco_ping emsserver` command.
 - d. Run the `nco_sql -u root -server emsserver` command.
7. Users connected to native desktops on an ObjectServer running on a UNIX server are complaining about slow response times. You suspect that a misconfigured automation has generated too many Journal records. What is the best technique for quickly verifying the total number of Journal records in the ObjectServer?
 - a. Enable ObjectServer profiling and check the ObjectServer log file for the total number of records.
 - b. Start a desktop client, highlight an event record, right-click the record, and select the Journal.
 - c. Start a desktop client, and modify the Metric setting on the All Events monitor box to show the total Journal records.

- d. Use the `nco_sql` utility to connect to the ObjectServer and issue the `select count(*) from alerts.journal; go` command.
8. Which configuration has the greatest effect on ObjectServer response time?
- Reaper frequency.
 - Granularity frequency.
 - Deduplication frequency.
 - Temporal trigger frequency.
9. What are two properties that need to be set to ensure failover functionality is enabled for the IBM Tivoli Netcool/OMNIBus Syslog probe? (Choose two.)
- PeerPort.
 - PeerHost.
 - SlavePort.
 - SlaveHost.
 - MasterHost.
10. Which rules file code segments shows the proper implementation of a multicolumn lookup table within a rules file?
- ```

table upsWellKnownAlarms =
{
 {"33.1.6.3.1","UPS Battery Status","4","100041"},
 {"33.1.6.3.5","UPS Temperature Status","4","100059"},
 {"33.1.6.3.6","UPS Input Status","4","4001"}
}
default = {"UPS Status","4","4001"}

```
  - ```

lookup upsWellKnownAlarms =
{
  {"33.1.6.3.1","UPS Battery Status","4","100041"},
  {"33.1.6.3.5","UPS Temperature Status","4","100059"},
  {"33.1.6.3.6","UPS Input Status","4","4001"}
}
default = {"UPS Status","4","4001"}

```
 - ```

33.1.6.3.1<tab>UPS Battery Status<tab>4<tab>100041
33.1.6.3.5<tab>UPS Temperature Status<tab>4<tab>100059
33.1.6.3.6<tab>UPS Input Status<tab>4<tab>4001

```
  - ```

33.1.6.3.1,UPS Battery Status,4,100041
33.1.6.3.5,UPS Temperature Status,4,100059
33.1.6.3.6,UPS Input Status,4,4001

```

Answer key

Here are the answers:

1. A
2. C
3. B
4. C
5. C
6. B
7. D
8. D
9. A and B
10. A

Archived

Archived

Abbreviations and acronyms

AEL	Active Event List	SME	Subject Matter Expert
AEN	Accelerated Event Notification	SNMP	Simple Network Management Protocol
AIX	Advanced Interactive eXecutive	SOA	Service-Oriented Architecture
API	Application Programming Interface	SQL	Structured Query Language
CGI	Common Gateway Interface	SSL	Secured Socket Layer
CPU	Central Processing Unit	SVI	Supervisory Instruction
CRM	Customer Relationship Management	TBSM	Tivoli Business Service Manager
DB2	Database 2	TCP/IP	Transaction Control Protocol/Internet Protocol
DBA	Database Administrator	UPS	Uninterruptible Power Supply
DNS	Domain Name Service	URL	Universal Resource Locator
EMS	Element Management System	VAP	Value Advantage Plus
ENMS	Enterprise Network Management System	VUE	Virtual University Enterprises
GMT	Greenwich Meridian Time	WAN	Wide Area Network
GUI	Graphical User Interface	XML	eXtensible Markup Language
HTML	Hyper Text Markup Language		
HTTP	Hyper Text Transfer Protocol		
I/O	Input/Output		
IBM	International Business Machines Corporation		
IDUC	Insert Delete Update Change		
ISQL	Interactive SQL		
ITIL	IT Infrastructure Library®		
ITSO	International Technical Support Organization		
JMS	Java Messaging System		
JVM	Java Virtual Machine		
ODBC	Open Database Connectivity		
ROI	Return on Investment		
SAF	Security		

Archived

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 196. Note that some of the documents referenced here may be available in softcopy only.

- ▶ *Best Practices for IBM Tivoli Enterprise Console to Netcool/OMNIBus Upgrade*, SG24-7557
- ▶ *Creating EIF Events with Tivoli Directory Integrator for Tivoli Netcool/OMNIBus and Tivoli Enterprise Console*, REDP-4352

Other publications

These publications are also relevant as further information sources:

- ▶ *IBM Tivoli Monitoring for Tivoli Netcool/OMNIBus User's Guide*, SC23-6375
- ▶ *IBM Tivoli Netcool/OMNIBus Administration Guide*, SC23-6371
- ▶ *IBM Tivoli Netcool/OMNIBus Installation and Deployment Guide*, SC23-6370
- ▶ *IBM Tivoli Netcool/OMNIBus Probe and Gateway Guide*, SC23-6373
- ▶ *IBM Tivoli Netcool/OMNIBus Release Notes*, GC23-6374
- ▶ *IBM Tivoli Netcool/OMNIBus User's Guide*, SC23-6372

Online resources

These Web sites are also relevant as further information sources:

- ▶ IBM Certification page:
<http://www.ibm.com/certify/index.shtml>

- ▶ IBM Test Objectives for Test 933:
<http://www-03.ibm.com/certify/tests/obj933.shtml>
- ▶ OMNibus FixPack download:
<http://www-01.ibm.com/software/sysmgmt/products/support/F495033D98219V85-download.html>
- ▶ Passport Advantage:
http://www-01.ibm.com/software/howtobuy/passportadvantage/pao_customers.htm

How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks publications, 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

Accelerated Event Notification, see AEN
administration console 40
Administrator 43
AEN 22
agent code 171
AlertsGateway 42
AlertsProbe 42
AlertsUser 42
ALTER command 76
AutoAdmin 42

B

bi-directional gateway 155
BREAK command 142

C

CatalogUser 41
certification
 achievements 26
 benefits 3
 checklist 5
 objectives 8
 overview 1
 program 2
 recommended study resources 35
class creation 143
commands
 ALTER 76
 BREAK 142
 EVTFT 164
 EXECUTE 142
 IDUC 164
 INSTALL 62
 isql 41, 171
 kill 150
 nco 105
 nco_aen 159
 nco_confpack 66
 nco_dbinit 69, 155, 157
 nco_event 70, 181
 nco_g_objserv_uni 97

nco_igen 96, 102, 155, 158
nco_objserv 70
nco_p_simnet 145, 174
nco_p_syntax 153
nco_pa_start 106
nco_pa_status 103–104
nco_pad 102
nco_patch 65
nco_ping 180
nco_sql 40, 56, 67, 97, 141
nco_xigen 74
netstat 73, 179
ps 145, 179
SNDMSG 164
telnet 180
top 184
traceroute 180
vmstat 184
components
 administrative 40
 architectural 38
 desktop 40
 IDUC 44
configuration
 export and import 66
 tiered 48
CRM 39
Customer Relationship Management, see CRM

D

database field definition 128
database tables 40
database triggers 40
DatabaseAdmin 42
Desktop Object Server 47
DesktopAdmin 42–43
directory ownership 69

E

environment variables 68
event filter 121
EVTFT command 164
EXECUTE command 142

F

failover 45
failover operation 176
filters definition 120
firewall consideration 52

G

gateway 39, 43
Gateway configuration 91
group creation 112
groups 41
growth through skills 6

H

health monitoring 171
Health Monitoring agent 171

I

IBM Professional Certification Program 2
IBM Tivoli Monitoring 171
IDUC 44
IDUC command 164
INSTALL command 62
installation
 fixpack 66
 process 62
ISQL 42–43
isql command 41, 171
ISQLWrite 43

K

kill command 150

M

menu creation 124
monitors 39

N

naming convention 50
nco command 105
nco_aen command 159
nco_confpack command 66
nco_dbinit command 69, 155, 157
nco_event command 70, 181
nco_g_objserv_uni command 97
nco_igen command 96, 102, 155, 158

nco_objserv command 70
nco_p_simnet command 145, 174
nco_p_syntax command 153
nco_pa_start command 106
nco_pa_status command 103–104
nco_pad command 102
nco_patch command 65
nco_ping command 180
nco_sql command 40, 56, 67, 97, 141
nco_xigen command 74
netstat command 73, 179
NO 171
Normal 43

O

objectives 8
ObjectServer 39
 communication 72
 creation 69
 properties 76
 verification 69
ObjectServer customization 117
ObjectServer tables 52
OMNibus
 problem determination 178
 tables 52

P

performance tuning 184
planning issues 49
port considerations 52
post implementation customization 68
probe 39, 43
probe configuration 144
probe features 49
Probe property 146
Probe rule file 150
problem determination 178
 desktop startup 181
 probe connection 180
 slow response time 181
process agent configuration 100
ps command 145, 179
Public 43

R

Redbooks Web site 196

Contact us xx
remote desktop installation 86
restriction filter 123
return on investment, see ROI
ROI 5
role creation 107
roles 41

S

security 41
security configuration 106
SecurityAdmin 42
simnet
 device definition 174
simnet probe 65
SME 8
SNDMSG command 164
Software Value Incentive, see SVI
SQL 41
SSL usage 50
startup problems 178
startup script 104
Structured Query Language, see SQL
study resources 35
Subject Matter Expert, see SME
SuperUser 43
SVI 7–8
System 43

T

telnet command 180
tiered implementation 48
Tivoli Software Professional Certification 4
tool definition 133
ToolsAdmin 42
top command 184
traceroute command 180
trigger creation 140

U

uni-directional gateway 157
user creation 115
users 41

V

Value Advantage Plus, see VAP
VAP 7–8

view creation 117
Virtual University Enterprises, see VUE
vmstat command 184
VUE 5

Archived



Certification Guide Series: IBM Tivoli Netcool/OMNIBus V7.2 Implementation



Detailed architecture and components discussion

This IBM Redbooks publication is a study guide for the IBM Tivoli Netcool/OMNIBus V7.2 implementation certification test. It is aimed at the IT professional who wants to be an IBM® Certified Professional for this product.

Installation and configuration processing

The IBM Tivoli Netcool/OMNIBus V7.2 certification test is offered through the IBM Professional Certification program. It is designed to validate the skills required of technical professionals who work in the implementation and deployment of IBM Tivoli Netcool/OMNIBus V7.2.

Event collection and automation

This book provides the necessary information for understanding the subject matter. It includes sample questions, which will help you evaluate your progress. It familiarizes the readers with the types of questions that may be encountered in the exam.

This guide does not replace practical experience, and it is not designed to be a stand-alone guide on this subject. Instead, this guide should be combined with educational activities and experiences and used as a useful preparation guide for the exam.

For your convenience, the chapters are based on the certification objectives of the IBM Tivoli Netcool/OMNIBus V7.2 implementation certification test. Those requirements are planning, prerequisites, installation, configuration, administration, and problem determination. Studying these chapters helps you prepare for the objectives of the exam.

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