ActiveBatch® User's Guide

Version 3

The Cross-Platform Enterprise Job Management and Scheduling System

Advanced Systems Concepts, Inc.

Sixth Edition: January 2002
Fifth Edition: August 2001
Fourth Edition: May 2001
Third Edition: April 2001
Second Printing: February 2001
First Printing: December 2000

Table of Contents

1 LICENSE AGREEMENTS **1**

1.1 EVALUATION LICENSE AGREEMENT 1
1.2 STANDARD LICENSE AGREEMENT 3

2 INTRODUCTION **7**

2.1 PRODUCT COMPONENTS 8
2.2 PRODUCT EDITIONS 9
2.3 REGISTRATION AND SUPPORT 11
2.3.1 Support Programs 12
2.3.1.1 Product Warranty 12
2.3.1.2 Post-Warranty Support 12

3 INSTALLATION **13**

3.1 PREREQUISITE HARDWARE/SOFTWARE 14
3.1.1 Windows 2000/NT/Me/9x Installation Process 16
3.2 MANAGING ACTIVEBATCH LICENSES 26
3.2.1 ActiveBatch Registry Information 28
3.3 STARTING ACTIVEBATCH 30
3.3.1 Job Scheduler Service Security 31
3.3.2 Job Scheduler E-Mail Facility 31
3.4 STOPPING ACTIVEBATCH SERVICES 32
3.5 MONITORING ACTIVEBATCH PERFORMANCE 33
3.6 ACTIVEBATCH DBPURGE 36
3.7 MIGRATING FROM BQMS 37
3.7.1 BQMS Command Compatibility 38
3.8 NON-MICROSOFT EXECUTION AGENTS 39
3.8.1 Compaq TRU64 Unix 39
3.8.1.1 Installation 39
3.8.1.2 DCE/RPC Configuration 39
3.8.1.3 Execution Agent Configuration 39
3.8.1.4 Startup and Shutdown 39
3.8.1.5 Batch Job Environment and Considerations 40
3.8.2 Sun Solaris 40
3.8.2.1 Installation 40
3.8.2.2 DCE/RPC Configuration 40
3.8.2.3 Execution Agent Configuration 40
3.8.2.4 Startup and Shutdown 41
3.8.2.5 Batch Job Environment and Considerations 41
3.8.3 Compaq OpenVMS 42
3.8.3.1 Installation 42
3.8.3.2 DCE/RPC Configuration 42

3.8.3.3 Execution Agent Configuration 42
3.8.3.4 Startup and Shutdown 43
3.8.3.5 Batch Job Environment and Considerations 43

4 ACTIVEBATCH TERMS AND CONCEPTS 45

4.1 ARCHITECTURE 46
4.2 OBJECTS 47
4.2.1 Job Object 47
4.2.2 Queue Object 47
4.2.3 Schedule/Calendar Objects 48
4.2.4 Event Object 48

5 QUICK TOUR (GETTING STARTED) 51

5.1 STARTING THE ACTIVEBATCH ADMIN APPLICATION 51
5.1.1 Introductory Assistance Wizard 52
5.2 GETTING STARTED TOUR 54
5.3 CREATING AN EXECUTION QUEUE 57
5.4 SUBMITTING A SIMPLE JOB 62
5.5 SUBMITTING A SCHEDULED JOB 68
5.6 USING THE "VIEWS" 73

6 USING ACTIVEBATCH WITH ABATADMIN 77

6.1 ABATADMIN TERMS AND CONCEPTS 78
6.1.1 Bars and Views 79
6.1.2 ActiveBatch Neighborhood View 79
6.1.3 ActiveBatch Icons 84
6.1.4 ActiveBatch Hot-Keys 87
6.1.5 Runbook View 88
6.1.6 Reporting Facility 89
6.1.6.1 Creating a Report 90
6.1.6.2 Exporting Report Information 92
6.1.7 Tools and Options 93
6.1.8 ABATAdmin and WMI 97
6.2 JOB SCHEDULER 98
6.2.1 Connecting to a Job Scheduler 98
6.2.2 Disconnecting from a Job Scheduler 100
6.2.3 Job Scheduler Properties 101
6.2.4 Managing Job Schedules 106
6.2.5 Managing Business Calendars 106
6.3 CREATING QUEUES 109
6.3.1 Creating an Execution Queue 109
6.3.1.1 New Execution/Generic Queue – General 110
6.3.1.2 New Execution Queue – Properties 112

6.3.1.3 New Execution/Generic Queue – Security 114
6.3.1.4 Add User Characteristic (Queue) 116
6.3.1.5 Add Security Dialog (Queue) 117
6.3.2 Creating a Generic Queue 118
6.4 MANAGING QUEUES 120
6.4.1 Modifying a Queue 120
6.4.2 Queue Counters 121
6.4.3 Deleting a Queue 122
6.4.4 Starting and Stopping a Queue 123
6.4.5 Open and Close a Queue 125
6.4.6 Flush a Queue 127
6.4.7 Associating/Disassociating a Member Execution Queue 128
6.5 JOB DEFINITION 130
6.5.1 Batch Job Security Requirements 130
6.5.2 Batch Job Files and Run-Time Considerations 132
6.5.3 Submitting a Job 133
6.5.3.1 New Job – General 134
6.5.3.2 New Job – Scheduling 138
6.5.3.3 New Job – Characteristics 141
6.5.3.4 New Job – Job Dependency 143
6.5.3.5 New Job – Execution 145
6.5.3.6 New Job – Logging/Alerts 149
6.5.3.7 New Job – Completion 151
6.5.3.8 New Job – Monitor/Simulation 154
6.5.3.9 New Job – Security 157
6.5.3.10 Job – Job History Dialog 159
6.5.3.11 Job – Add Security Dialog 163
6.5.3.12 Job – Machine Characteristics 165
6.5.3.13 Job – Select User Characteristics 167
6.5.3.14 Job – Audit Trail 168
6.5.4 Job Schedules 169
6.5.4.1 Create a Schedule 170
6.5.4.2 Copying a Schedule 174
6.5.4.3 Editing a Schedule 174
6.5.4.4 Deleting a Schedule 174
6.5.4.5 Associating a Schedule 175
6.5.4.6 Disassociating a Schedule 175
6.5.4.7 Securing a Schedule/Business Calendar 176
6.5.5 Job Dependencies 178
6.5.5.1 Adding a Job Dependency 180
6.5.6 Event Triggers 181
6.5.7 Success/Failure Batch Start 185
6.5.8 Job Notifications 186
6.5.9 Job Alerts 187
6.5.10 Modifying/Viewing a Job 189
6.5.11 Abort/Delete a Job 191

6.5.12 Hold and Release a Job 193
6.5.13 Pause and Resume a Job 194
6.5.14 Restart Job Execution 195
6.5.15 Resubmit Job 195
6.5.16 Trigger Job Execution 196
6.5.17 Copy and Paste a Job 198
6.5.18 Moving a Job 199
6.6 ONDEMAND QUEUE 200
6.6.1 Security Implications 200
6.6.1.1 Trigger Access 200
6.6.1.2 Trigger with Queue Change Access 201
6.6.1.3 Trigger with Change Parameters Access 201
6.6.2 Second Party Job Release 201
6.7 JOB CHECKPOINT AND RESTART 202

7 ACTIVEBATCH FEATURES AND THEIR APPLICATIONS 203

7.1 ADVANCED CONNECTIONS 203
7.2 SETTING UP YOUR QUEUES 204
7.2.1 Execution Queues 204
7.2.2 Generic Queues 205
7.2.2.1 Characteristics 206
7.2.2.2 Queue Scheduling Algorithms 208
7.2.2.3 Job Broadcast 209
7.3 SETTING UP YOUR JOBS 210
7.3.1 Job Property "Minimums" 210
7.3.2 Success Code Rule 211
7.3.3 Triggers 212
7.3.3.1 Date and Time Triggers 212
7.3.3.2 Business Date Processing (Calendars) 221
7.3.3.3 Events 223
7.3.3.4 ActiveBatch Trigger 228
7.3.4 Pre and Post Job Steps 228
7.3.5 Job Dependencies 230
7.3.6 Batch Starts 233
7.3.7 Multi-Processor Affinity 234
7.3.8 Job Operational Features 235
7.3.9 Notifications and Alerts 236
7.3.10 Job Monitoring 237
7.3.11 Restart and Job Checkpointing 238
7.3.11.1 Failover 238
7.3.11.2 On Reboot 238
7.3.11.3 On Process Failure 239
7.3.11.4 Checkpoint 239
7.4 TROUBLESHOOTING 240

7.4.1 Job Output Logging *240*
7.4.1.1 User Managed Logging 240
7.4.1.2 Centrally Managed Logging 241
7.4.2 Job History *242*
7.4.2.1 Audits 244
7.4.2.2 History 245
7.5 SECURITY 246

8 ADMINISTERING ACTIVEBATCH THROUGH MMC 247

8.1 ADDING THE SNAP-IN 248
8.1.1 Using the ActiveBatch MSC Snap-In *248*
8.1.2 Manually Adding the Snap-In *249*
8.2 ACTIVEBATCH SERVICES 253
8.3 ACTIVEBATCH REGISTRY ENTRIES 255
8.4 ACTIVEBATCH POLICY MANAGEMENT 257
8.4.1 System Policies *259*
8.4.2 Queue Policies *260*
8.4.3 Job Policies *261*
8.5 ASSIGNING DEFAULT OBJECT SECURITY 263
8.6 ACTIVEBATCH ADMINISTRATORS 264

9 USING THE ACTIVEBATCH COMMAND-LINE INTERFACE 267

9.1 CONVENTIONS 268
9.1.1 Absolute Time *268*
9.1.2 Delta Time *269*
9.1.3 Combination Time *270*
9.2 COMMANDS 271
9.2.1 CLOSE/QUEUE *271*
9.2.2 DELETE/JOB *272*
9.2.3 DELETE/QUEUE *273*
9.2.4 FLUSH/QUEUE *274*
9.2.5 HOLD *275*
9.2.6 INITIALIZE Queue *276*
9.2.7 MODIFY/JOB *279*
9.2.8 MODIFY/QUEUE *285*
9.2.9 OPEN/QUEUE *288*
9.2.10 PAUSE *289*
9.2.11 RELEASE *290*
9.2.12 REQUEUE *291*
9.2.13 RESTART *292*
9.2.14 RESUME *293*
9.2.15 SET BATCH *294*
9.2.16 SHOW JOB *296*
9.2.17 SHOW QUEUE *298*

9.2.18 START/QUEUE 300
9.2.19 STOP/QUEUE 301
9.2.20 SUBMIT 302
9.2.21 SYNCHRONIZE 308
9.2.22 TRIGGER 309

10 ACTIVEBATCH SCRIPTING AND PROGRAMMING 311

10.1 ACTIVEBATCH OBJECT TOPOLOGY 312

11 ACTIVEBATCH MESSAGES 313

12 INDEX 335

Table of Figures

Figure 1 - Type of Installation (Windows 2000/NT) 16
Figure 2 - Customer Information (Windows 2000/NT/9x/Me) 17
Figure 3 - Product Location Information (Windows 2000/NT/9x/Me) 18
Figure 4 - Component Selection (Windows 2000/NT/9x/Me) 19
Figure 5 - Job Scheduler Data Path Dialog (Windows 2000/NT) 20
Figure 6 – Job Scheduler SMTP Settings (Windows 2000/NT) 21
Figure 7 - Execution Agent Option Dialog (Windows 2000/NT) 22
Figure 8 - Select Program Folder (Windows 2000/NT/9x/Me) 23
Figure 9 - Confirm Final Settings (Windows 2000/NT/9x/Me) 24
Figure 10 - Working with ABATLIC 27
Figure 11 - Job Scheduler/Execution Agent NT Services 30
Figure 12 - Add to Chart "Job Scheduler" 33
Figure 13 - Add to Chart "Queue" 34
Figure 14 - ActiveBatch Performance Monitoring 35
Figure 15 - ActiveBatch Architecture 46
Figure 16 - ActiveBatch Admin Utility (Initial) 51
Figure 17 - Introductory Assistance Wizard - Startup 52
Figure 18 - Introductory Assistance Wizard - Job Scheduler Connect 53
Figure 19 - ABATADmin "Connect" dialog 54
Figure 20 - Connect (Advanced) Dialog 55

Figure 21 - Quick Tour Sample Connection 56
Figure 22 - New Queue Wizard – Page 1 57
Figure 23 - New Queue Wizard – Page 2 58
Figure 24 - New Queue Wizard – Page 3 59
Figure 25 - New Queue Wizard – Last Page 60
Figure 26 - Quick Tour New Queue Wizard 61
Figure 27 - New Job Wizard – Page 1...62
Figure 28 - New Job Wizard – Page 2...64
Figure 29 - New Job Wizard – Page 3...65
Figure 30 - New Job Wizard – Page 4...66
Figure 31 - New Job Wizard – Last Page 67
Figure 32 - New Job Wizard – Recurring/Immediate 68
Figure 33 - New Job Wizard – Schedule Dialog 69
Figure 34 - New Job Wizard – Schedule Create 70
Figure 35 - Job Wizard Notification Page 71
Figure 36 - Quick Tour Delete Job Message Box 72
Figure 37 - Quick Tour Runbook View (Today/1Day) 73
Figure 38 - Quick Tour Sample Job Tabs 74
Figure 39 - Quick Tour Job History Tab 75
Figure 40 - Quick Tour Job Audit Trail 76
Figure 41 - ABATAdmin Initial Usage...78
Figure 42 - Navigation Bar – Job Scheduler Tab 80
Figure 43 - Navigation Bar – Execution Agents Tab 81
Figure 44 - Navigation Bar – Index Tab 82
Figure 45 - Selection and Right-Click Menu...83
Figure 46 - Navigation View (Details)..84
Figure 47 - Tool Bar...86
Figure 48 - Runbook Main Window (1 Day) 88
Figure 49 - Reporting Facility 89
Figure 50 - Report Filter Dialog..90
Figure 51 - Report Filter Add Dialog..91
Figure 52 - Exporting Data..92
Figure 53 - Options (General) 93
Figure 54 - Options (Runbook) 94
Figure 55 - Options (Reporting)..95
Figure 56 - Options (Filters)..96

Figure 57 - Connect Dialog (Basic)....................98
Figure 58 - Connect Dialog (Advanced) 99
Figure 59 - Job Scheduler Properties....................101
Figure 60 - Queue Policies (Job Scheduler)....................102
Figure 61 - Job Policies (Job Scheduler)....................103
Figure 62 - Job Scheduler Default Security....................104
Figure 63 - Job Scheduler Performance Counters....................105
Figure 64 - Defining a Business Calendar....................107
Figure 65 - New Execution/Generic Queue (General) 110
Figure 66 - New Execution Queue (Properties) 112
Figure 67 - Queue Security Tab 115
Figure 68 - New Execution Queue (Add User Characteristic)....................116
Figure 69 - Add Security Dialog (Queue) 117
Figure 70 - New Generic Queue (Property) 119
Figure 71 - Queue Counters 121
Figure 72 - Delete Queue Confirmation....................122
Figure 73 – Stopping/Starting a Queue 123
Figure 74 - Stopping a Queue (Timed)....................124
Figure 75 - Opening or Closing a Queue....................125
Figure 76 - Restrict Job Submission (Timed)....................126
Figure 77 - Flush a Queue....................127
Figure 78 – Associating/Disassociating an Execution Queue Dialog 128
Figure 79 - Adding the BatchLogon Right....................131
Figure 80 - New Job Property Tabs....................133
Figure 81 - New Job – General 134
Figure 82 - New Job – Scheduling....................138
Figure 83 - New Job – Characteristics 141
Figure 84 - New Job -- Job Dependency Dialog 143
Figure 85 - New Job -- Execution Dialog....................145
Figure 86 - New Job – Log/Alerts....................149
Figure 87 - New Job – Completion 151
Figure 88 - New Job -- Monitor/Simulation Dialog....................154
Figure 89 - New Job -- Security Dialog 157
Figure 90 - Job History (Instance)....................159
Figure 91 - Job History (Template)....................162

Figure 92 - Job Add Security Dialog Box 163
Figure 93 - Machine Characteristic Dialog 165
Figure 94 - Select User Characteristics 167
Figure 95 - Job Audits 168
Figure 96 - Schedule List Dialog 169
Figure 97 - Create a Schedule Dialog 171
Figure 98 - Select a Schedule Dialog 175
Figure 99 - Schedule/Business Calendar Security Dialog 176
Figure 100 - Job Dependency Tab Dialog 178
Figure 101 - Adding a Job Dependency Dialog 180
Figure 102 - Event Trigger Dialog 181
Figure 103 - WMI Event Privilege Settings 183
Figure 104 - Event Advanced Settings 184
Figure 105 - Batch Start Dialog 185
Figure 106 - Job Notification Dialog 186
Figure 107 - Job Alert Dialog 187
Figure 108 - Modifying a Job 189
Figure 109 - Job Modification Confirmation 190
Figure 110 – Abort/Delete a Job Dialog 191
Figure 111 -- Abort/Delete a Template Job Dialog 191
Figure 112 -- Delete Template Confirmation Dialog 192
Figure 113 - Hold/Release a Job 193
Figure 114 - Pause/Resume a Job 194
Figure 115 - Job Resubmit/Queue Selection 195
Figure 116 - Triggering a Job 196
Figure 117 - Trigger Job Dialog 197
Figure 118 - Moving a Job 199
Figure 119 - Execution Queue Properties 204
Figure 120 - Job Broadcast 209
Figure 121 - MMC Console 248
Figure 122 - Add ActiveBatch Snap-in 249
Figure 123 - MMC ActiveBatch Neighborhood 250
Figure 124 - MMC Connect Dialog 251
Figure 125 - MMC Properties 252
Figure 126 - MMC ActiveBatch Services 254
Figure 127 - MMC Registry Entries 256

Figure 128 - MMC Policies........257
Figure 129 - MMC Modify Policy........258
Figure 130 - MMC Default Job Security........263
Figure 131 - MMC System Security........265
Figure 132 - ActiveBatch Object Relationship........312

Tables

Table 1 - Product Editions Feature Summary........10
Table 2 - Queue Security Access........115
Table 3 - Operating System Batch Job Considerations........132
Table 4 - Job States and Sub-States........136
Table 5 - Environment Variables for Job Steps........146
Table 6 - Job Security Access........158
Table 7 - Machine Characteristics........166
Table 8 - Schedule/Business Calendar Access........177
Table 9 - Job Alerts........188
Table 10 - System Policies........259
Table 11 - Queue Policies........260
Table 12 - Job Policies........262
Table 13 - ActiveBatch Messages (Alphabetical)........323
Table 14 - ActiveBatch Messages (Numeric)........334

1 LICENSE AGREEMENTS

The sections that follow contain two (2) License Agreements. The first agreement governs use of the ActiveBatch product when the software is being evaluated as part of either the self-generated evaluation serial number or an evaluation serial number produced by Advanced Systems Concepts, Inc. or one of its distributors. Product Evaluations are always time-limited (typically 30 days but for a period not to exceed 180 days). The second agreement governs use of the ActiveBatch product when the software has been purchased for use. Unless a written agreement between the User and Advanced Systems governing use of the Software has been executed, one of the agreements below will govern based on purchase or evaluation. The User must agree to the License Terms and Conditions in order to properly use the software.

ActiveBatchTM V3

The sections that follow are the license agreements that governs this product. A separate license agreement also accompanies this document and product. Do not break open the CD case or the envelope containing the product serial number unless you agree to the terms of this license agreement and you have decided to retain the product. **Once the serial number envelope has been opened, the software product cannot be returned.**

1.1 Evaluation License Agreement

The software which accompanies this license (the "Software") is the property of Advanced Systems Concepts, Inc. or its licensors and is protected by copyright law. While Advanced Systems Concepts, Inc. continues to own the Software, you will have certain rights to use the Software after your acceptance of this license. Your rights and obligations with respect to the use of this Software are as follows:

You may make as many copies of this EVALUATION Software and run the Software on as many systems as you wish. However, when the Software is purchased, this license will NOT APPLY to that Software and the Standard License Agreement (see Section 1.2) will be in effect.

You acknowledge and understand that this EVALUATION software will expire and become unusable after a certain date. You agree not to modify or change the Software, the Software's Serial Number or your computer system to otherwise disable or avoid the expiration date of the Software. The expiration date is displayed as part of the Software startup process each time the Software is executed. To continue operation of the Software product you will be required to purchase the current release of the Software and agree to the Software license agreement that governs that Software's use. However, you are under no obligation to purchase the Software product as a result of evaluating this product. If you elect not to purchase the Software you will be required to uninstall the Software product.

You may not:

(i) copy printed materials which accompanies the Software;

(ii) reverse engineer, decompile, disassemble, modify, translate, make any attempt to discover the source code of the Software, or create derivative works from the Software; or
(iii) change or otherwise disclose the Evaluation Serial Number provided for evaluating the Software to anyone outside your company or organization

ADVANCED SYSTEMS CONCEPTS DOES NOT WARRANT THIS EVALUATION SOFTWARE PRODUCT. USER ACCEPTS THAT SOFTWARE MAY CONTAINS BUGS AND OTHER SOFTWARE WHICH MAY BE NON-FUNCTIONAL AND THAT ALL SOFTWARE IS ACCEPTED ON AN "AS-IS" BASIS.

Disclaimer of Damages:

REGARDLESS OF WHETHER ANY REMEDY SET FORTH HEREIN FAILS OF ITS ESSENTIAL PURPOSE, IN NO EVENT WILL ADVANCED SYSTEMS CONCEPTS, INC. BE LIABLE TO YOU FOR ANY SPECIAL CONSEQUENTIAL, INDIRECT OR SIMILAR DAMAGES, INCLUDING ANY LOST PROFITS OR LOST DATA ARISING OUT OF THE USE OR INABILITY TO USE THE SOFTWARE EVEN IF ADVANCED SYSTEMS CONCEPTS, INC. HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

SOME STATES DO NOT ALLOW THE LIMITATION OR EXCLUSION OF LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES SO THE ABOVE LIMITATION OR EXCLUSION MAY NOT APPLY TO YOU.

IN NO SUCH CASE SHALL ADVANCED SYSTEMS CONCEPTS, INC. LIABILITY EXCEED THE PURCHASE PRICE FOR THE SOFTWARE. The disclaimers and limitations set forth above will apply regardless of whether you accept the Software.

U.S. Government Restricted Rights:

RESTRICTED RIGHTS LEGEND. Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c) (1) (ii) of the Rights in Technical Data and Computer Software clauses at DFARS 252.227-7013 or subparagraphs (c) (1) and (2) of the Commercial Computer Software-Restricted Rights clause at 48 CFR 52.227-19, as applicable, Advanced Systems Concepts, Inc., 33-41 Newark Street, Hoboken, NJ 07030-5604.

General:

This Agreement will be governed by the laws of the State of New Jersey. This Agreement may only be modified by a license addendum which accompanies this license or by a written document which has been signed by both you and Advanced Systems Concepts, Inc.

For more information about ASCI's licensing policies, please call ASCI Product Licensing at (201) 798-6400 and press 181 or via facsimile at (201) 798-9203 or via e-mail at: admin@advsyscon.com.

1.2 Standard License Agreement

IMPORTANT--PLEASE READ CAREFULLY. THIS IS A LEGAL AGREEMENT BETWEEN YOU (EITHER AN INDIVIDUAL OR AN ORGANIZATION) AND ADVANCED SYSTEMS CONCEPTS, INC., FOR THE SOFTWARE PRODUCT IDENTIFIED ABOVE, WHICH INCLUDES COMPUTER SOFTWARE AND RELATED DOCUMENTATION. BY BREAKING THE ENVELOPE SEAL CONTAINING THE SERIAL NUMBER OR INSTALLING, OR OTHERWISE USING THE SOFTWARE, YOU ACCEPT ALL THE TERMS AND CONDITIONS OF THIS AGREEMENT. IF YOU DO NOT AGREE TO THESE TERMS, YOU MAY, WITHIN THIRTY (30) DAYS OF PURCHASE, RETURN THE UNUSED SOFTWARE PRODUCT TO THE LOCATION WHERE YOU OBTAINED IT FOR A REFUND, SUBJECT TO A RESTOCKING FEE.

Advanced Systems Concepts, Inc., a New Jersey Corporation, whose address is 33-41 Newark Street, Hoboken, New Jersey, 07030, USA ("ASCI") hereby grants to you a non-exclusive license (a "License") to use the software identified above (the "Software") and the accompanying printed materials and User Manual (the "Documentation") on the terms set forth below.

1. **GRANT OF LICENSE.** This license grants you the following rights.

- **Software.** Except as set forth below, you may use the Software on a single computer. This license is applicable to all ActiveBatch Product Editions. Your right to use the Software is limited to the version you purchased and no rights are granted to any new versions of Software.

- **Storage/Network Use.** You may store or install a copy of the Software on a storage device, such as a network server, used only to install or run the Software on your computers over an internal network; however, you must acquire and dedicate a license for each separate computer on which the Software is installed or run from the storage device. A license for the Software may not be shared or used concurrently on different computers. You agree to inform all users of the Software of the terms and conditions for its use.

- **License Pack.** If you have acquired this Agreement as part of a License Pack, then you may make the number of additional copies of the Software as authorized on the printed copy of that License Pack, and you may use each copy in the manner described above

- **Upgrades.** When a Software Upgrade is purchased it must be applied to a previously licensed and installed Software product on a single computer. Purchase of a Software Upgrade does not constitute the right to use the Software on another computer.

- **Transfer.** You may transfer the Software and Documentation to a single recipient on a permanent basis provided you retain no copies of the Software or Documentation (including backup or archival copies) and the recipient agrees to the terms and conditions of this Agreement. If the Software is an upgrade, any transfer must include all prior versions of the Software and Documentation.

- **Dual Media.** If this Software was delivered on more than one form of electronic media, the right to use the Software is still limited to a single computer.

2. OTHER RIGHTS AND LIMITATIONS.

- **Licensing Management.** This Software has the ability to regulate its use based on this Agreement and employs a License Management Facility. You understand and acknowledge that the ability to copy the Software and/or to breach the Agreement may be prevented through the use of this License Management Facility. If you purchase a Software Upgrade, you acknowledge that the Software Serial Number contains special logic to ensure that a previous version of the Software Serial Number is requested and validated for installation. Further you agree to keep the Software Serial Number confidential and not to disclose the serial number to anyone outside your Company. You understand and acknowledge that the various ActiveBatch Product Editions provide varying access to the underlying ActiveBatch and other ASCI technologies. Any product limitations are noted within the Documentation.

- **Reverse Engineering.** You may not modify, translate, reverse engineer, de-compile, disassemble the Software, or make any attempt to discover the source code to the Software.

- **Copying.** You may not copy the Software or Documentation except as specifically provided by this Agreement.

- **Separation of Components.** The Software is licensed as a single product. You may not separate the Software's component parts for use on more than one computer.

- **Rental.** You may not rent or lease the Software or Documentation.

- **Proprietary Notices.** You may not remove or modify any proprietary notices, labels or marks on the Software or Documentation.

3. TITLE AND COPYRIGHT.

Title, ownership rights, and intellectual property rights in and to the Software and Documentation shall remain with ASCI. The Software and Documentation is protected by copyright laws of the United States and international copyright treaties. You may not copy any of the written materials accompanying the Software unless specifically authorized by ASCI in writing.

4. TERMINATION.

This License is in effect until terminated. The License will terminate automatically if you fail to comply with the limitations described herein. You may also voluntarily terminate the License. On termination, you must destroy all copies of the Software and Documentation.

5. EXPORT CONTROLS.

Since this Software is subject to the export control laws of the United States, you may not export or re-export the Software without the appropriate United States and foreign government licenses. You shall otherwise comply with all applicable export control laws and shall defend, indemnify and hold ASCI and all ASCI suppliers harmless from any claims arising out of your violation of such export control laws.

6. MISCELLANEOUS.

This Agreement represents the complete agreement concerning this license between the parties and supersedes all prior agreements and representations between them. This Agreement may be amended only in writing executed by both parties. THE ACCEPTANCE OF ANY PURCHASE ORDER PLACED BY YOU IS EXPRESSLY MADE CONDITIONAL ON YOUR ASSENT TO THE TERMS SET FORTH HEREIN, AND NOT THOSE CONTAINED IN YOUR PURCHASE ORDER. If any provisions of this Agreement is held to be unenforceable for any reason, such provision shall be reformed only to the extent necessary to make it enforceable and the remainder of this Agreement shall nonetheless remain in full force and effect. This Agreement shall be governed by and construed under New Jersey law as such law applies to agreements between New Jersey residents entered into and to be performed within New Jersey, except as governed by Federal law. The application of the United Nations Convention of Contracts for the International Sale of Goods is expressly excluded.

7. U.S. GOVERNMENT END-USERS

The Software and Documentation are provided with RESTRICTED RIGHTS. Use, duplication or disclosure by the U.S. Government is subject to restriction as set forth in subparagraph (c) (1)(ii) of The Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 or subparagraphs (c)(1) and (2) of the Commercial Computer-Restricted Rights at 48 CFR 52.227-19, as applicable. The contractor/manufacturer is Advanced Systems Concepts, Inc., 33-41 Newark Street, Hoboken, NJ, 07030, USA.

8. LIMITED WARRANTY

ASCI WARRANTS THAT THE MEDIA, IF PROVIDED BY ASCI, IS FREE FROM DEFECTS IN MATERIALS AND WORKMANSHIP FOR SIXTY (60) DAYS FROM THE DATE YOU ACQUIRE THE SOFTWARE. ASCI WARRANTS THAT THE SOFTWARE, AS PROVIDED BY ASCI, WILL PERFORM SUBSTANTIALLY ACCORDING TO THE ACCOMPANYING WRITTEN MATERIALS FOR SIXTY (60) DAYS FROM THE DATE YOU ACQUIRE THE SOFTWARE.

ASCI's sole liability, and your sole remedy, for any breach of this warranty shall be, in ASCI's sole discretion: (i) to replace your defective media; or (ii) if the above remedy is impractical, to refund the License fee received by ASCI for the Software. Replaced Software shall be covered by this limited warranty for the period remaining under the warranty that covered the original Software, or if longer, for thirty (30) days after the date of shipment to you of the replaced Software. Only if you inform ASCI of your problem with the Software during the applicable warranty period and provide evidence of the date you acquired the Software will ASCI be obligated to honor this warranty. ASCI will use reasonable commercial efforts to replace or refund pursuant to the foregoing warranty within thirty (30) days of being so notified.

THIS IS A LIMITED WARRANTY AND IT IS THE ONLY WARRANTY MADE BY ASCI. ASCI MAKES NO OTHER WARRANTY, REPRESENTATION OR CONDITION, EXPRESS OR IMPLIED, AND EXPRESSLY DISCLAIMS THE IMPLIED WARRANTIES OF MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. THE DURATION OF IMPLIED WARRANTIES OR CONDITIONS, INCLUDING WITHOUT LIMITATION, WARRANTIES OR CONDITIONS OF MECHANTABILITY AND OF FITNESS FOR A PARTICULAR PURPOSE, IS LIMITED TO THE ABOVE LIMITED WARRANTY PERIOD; SOME JURISDICTIONS DO NOT ALLOW LIMITATIONS ON HOW LONG AN IMPLIED WARRANTY OR CONDITION LASTS, SO LIMITATIONS MAY NOT APPLY TO YOU. NO ASCI AGENT, SUPPLIER OR EMPLOYEE IS AUTHORIZED TO MAKE ANY MODIFICATIONS, EXTENSIONS OR ADDITIONS TO THIS WARRANTY.

This Limited Warranty is void if failure of the Software and/or media, has resulted from accident, abuse, or improper use; if any modifications are made to the Software other than by ASCI; or if you violate the terms of this Agreement.

THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS, AND YOU MAY HAVE OTHER LEGAL RIGHTS THAT VARY FROM STATE TO STATE OR BY JURISDICTION.

9. LIMITATION OF LIABILITY.

UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, TORT, CONTRACT, OR OTHERWISE, SHALL ASCI OR ITS SUPPLIERS/RESELLERS/DISTRIBUTORS BE LIABLE TO YOU OR ANY OTHER PERSON FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES INCLUDING WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES, OR FOR ANY DAMAGES IN EXCESS OF ASCI'S LIST PRICE FOR A LICENSE TO THE SOFTWARE AND DOCUMENTATION, EVEN IF ASCI SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY. FURTHERMORE, SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS LIMITATION AND EXCLUSION MAY NOT APPLY TO YOU.

For more information about ASCI's licensing policies, please call ASCI Product Licensing at (201) 798-6400 and press 181 or via facsimile at (201) 798-9203 or via e-mail at: admin@advsyscon.com.

2 INTRODUCTION

ActiveBatch® is a distributed Job Scheduling and Management System that allows you to execute tasks in an unattended or background mode. With ActiveBatch you can:

- Define jobs with pre/post job steps and pre/post dependencies.
- Schedule these tasks or *jobs* on a date and time basis and/or based on system/application events.
- Execute a job on a specific machine or a class of machines.
- Execute a job on a virtual machine in which a specific machine is chosen based on workload or other characteristics.
- Execute a job on different hardware and software platforms.
- Obtain a complete audit trail of the submitted job including a log of the output of the job.
- Specify monitoring, alert and notification capability to ensure that jobs run properly and as expected.

2.1 Product Components

ActiveBatch consists of three (3) major components:

Client

As you would expect the Client components are the main user interaction point to the ActiveBatch system. ActiveBatch provides four (4) sets of clients depending on the type of user you are and what mode of interaction you require. Most users will execute the ActiveBatch Admin (ABATAdmin) utility. This Windows GUI-based component provides the user with the ability to create and manage job(s). ActiveBatch provides administrative users with a Microsoft Management Console (MMC) Snap-in to properly manage operations. ActiveBatch provides a complete set of command-line utilities so users can script ActiveBatch operations. As ActiveBatch is COM based, a programmatic interface is also available. Object-oriented scripting languages can directly access ActiveBatch's objects, properties and methods through the various constructs of those languages.

Job Scheduler

The ActiveBatch Job Scheduler is the heart of the job management system. Clients communicate with a Job Scheduler, which is responsible for creating all of the various objects that make up ActiveBatch. As its name implies, the Job Scheduler is the component that keeps track of all jobs and determines when a job is to execute. The Job Scheduler may be limited in terms of the number and type of Execution Agent(s) it can support. Any limitations are noted in the next section on Product Editions.

Execution Agent

Once a job is ready to execute, the Job Scheduler communicates with a machine's Execution Agent. This is the component that is responsible for actually running the job. The Execution Agent is the cross-platform component of the ActiveBatch system. Each Execution Agent is licensed for use. RPC/DCE is used for communications between the Job Scheduler and Execution Agents.

2.2 Product Editions

ActiveBatch is licensed in four (4) product edition packages: Personal Edition, Small Business Server, Advanced Server and Enterprise Cross-Platform Server. Each edition provides certain functionality and also varying restrictions.

Each product edition is packaged with one (1) Execution Agent. An Execution Agent is the ActiveBatch component that actually runs a job. The Execution Agent provided with each product edition is meant to be run on a Windows 2000 or Windows NT machine. Additional Execution Agents can be purchased separately.

The *Personal Edition* is a local-only version of ActiveBatch. All components of the product are expected to run and be accessed on a single machine. The Personal Edition may be used on a Windows 2000 Professional or Windows NT Workstation only.

The *Small-Business Server Edition* allows *access* to up to ten (10) Execution Agents. One Agent is included with the base product so up to nine (9) additional agents may be purchased and installed. The access restriction is determined by counting the number of Execution Queues directed to different machines. (In other words, this is not a "live" connection count but rather a simple static count of one Execution Machine to one Execution Agent). This edition further limits access to those Execution Agents that run on a Microsoft operating system.

The *Advanced Server Edition* allows for access to an unlimited number of Execution Agents. As noted you are provided with one (1) Execution Agent license as part of the base product. Additional Execution Agents can be purchased separately. This edition limits access to those Execution Agents that run on a Microsoft operating system.

The *Enterprise Cross-Platform Server Edition* allows for access to an unlimited number of Execution Agents. As noted you are provided with one (1) Execution Agent license as part of the base product. Additional Execution Agents can be purchased separately. This edition specifically supports both Microsoft and non-Microsoft based Execution Agents.

The Client software components included with ActiveBatch are available for Windows 2000, Windows NT (Intel) and Windows 9x/Me use. This software may be installed on as many machines as needed. Portions of the Client software only support Windows 2000 or Windows NT (see Section 3.1).

ActiveBatch Execution Agents (EA) are available at additional cost. EAs may be deployed on Windows 2000, Windows NT (Intel and Alpha), Compaq Tru64 UNIX, Sun Solaris (Sparc) UNIX and Compaq OpenVMS (check our web site at http://www.advsyscon.com for the latest updated information). Non-Microsoft based EAs may only be deployed and accessed after purchasing an Enterprise Cross-Platform Server Edition.

The table below summarizes the features and restrictions.

Feature	Personal Edition	Small Business Server	Advanced Server	Enterprise Cross Platform Server
Workstation	**Yes**	**Yes**	**Yes**	**Yes**
Server	**No**	**Yes**	**Yes**	**Yes**
Multiple Execution Agent Support	**No (Local Only)**	**Yes (Restricted to accessing up to ten (10) agents)**	**Yes**	**Yes**
Non-Microsoft-based Execution Agent Support	**No**	**No**	**No**	**Yes**
Unlimited Client Support	**Yes**	**Yes**	**Yes**	**Yes**

Table 1 - Product Editions Feature Summary

2.3 Registration and Support

Please take a moment and fill out the Product Registration Card found in your ActiveBatch package. In order to receive the benefits of the product support programs offered by Advanced Systems Concepts for ActiveBatch, we must know who you are.

Product Registration can be performed in one of several ways:

1. As part of a successful standard installation, ActiveBatch will ask whether you wish to connect to our web site and complete an online registration form. This is the simplest approach since some of the information can be provided in an automated manner. All information displayed is available for you to edit and confirm.

2. You can fill out the Product Registration Card and either send it by mail or FAX.

3. You can connect to our web site (www.advsyscon.com) at a time of your choosing and fill out the electronic version of the Registration Card (remember to have your serial number handy).

When you register your ActiveBatch license you may submit product support requests via E-mail (support@advsyscon.com) or FAX, 24 hours a day, 7 days a week as part of your product warranty.

2.3.1 Support Programs

Advanced Systems Concepts (ASCI) offers many support programs to assist ActiveBatch evaluators and customers. The following sections detail some of the features when registering your licensed usage with ASCI.

2.3.1.1 Product Warranty

Registered users of ActiveBatch are provided a sixty (60) day period in which an ASCI Support Engineer will be assigned to manage your technical question or problem. You may phone, FAX or E-mail your request to Advanced Systems Concepts. You will be asked for your product serial number (found on the Product Registration Card) when requesting assistance, so please keep it handy.

2.3.1.2 Post-Warranty Support

Advanced Systems offers several plans for ensuring that you always have assistance after the product warranty period has expired. ASCI provides two plans for annual product support: Priority Technical Support and Priority Technical Support Plus. These annual maintenance plans are designed to support and assist your use of ActiveBatch. ASCI also provides an annual plan where future upgrades and/or new versions are included. Please visit our web site for the last information and current prices on all these plans.

Most of these plans need to be purchased at the time you licensed the software, so please contact ASCI or your local distributor for more information.

3 INSTALLATION

This section describes the installation procedure when installing ActiveBatch from either a CD or a product download.

Insertion of the CD should cause an autorun of the ActiveBatch installation procedure. If your machine has autorun disabled look for the file named SETUP.EXE in folder [WIN2KNT] for Windows 2000/NT or [WIN9x] for Windows 9x/Me systems[1] and RUN that file. An ActiveBatch splash screen followed by information indicating that the Installation Wizard is initializing will be displayed. The standard Welcome dialog box appears reminding you that you are about to install all or some ActiveBatch components.

The installation process for Windows 2000/NT allows you to install all or selected ActiveBatch components. The installation process for Windows Me/9x only allows you to install all or selected Client components. The installation dialogues for all systems are quite similar.

ASCI highly recommends that you perform this installation on Windows 2000/NT using the Administrator account or a similar account with administrative rights. You must have administrative rights in order to install the Job Scheduler and/or Execution Agent components.

Note: ActiveBatch fully supports Microsoft's new Windows XP version in all variations. For simplicity whenever you see a Windows 2000 reference please consider that applicable to Windows XP as well.

[1] While you can successfully run the [WINNT9x] installation procedure on a Windows 2000 machine, only the [WIN2K] procedure is Microsoft BackOffice logo compliant.

3.1 Prerequisite Hardware/Software

Operating system software and hardware requirements vary depending on which components of ActiveBatch you choose to install. The product has no special installation memory requirements and the installation procedure requires between 2 and 8 megabytes of hard disk depending on Product Edition, components and platform selected.

Full or Job Scheduler Only: A complete installation of ActiveBatch (or a Job Scheduler only installation) is supported on Windows 2000 or Windows NT V4 (running at least Service Pack 4) on an Intel® compatible processor. ASCI recommends a minimum configuration of a Pentium 166Mhz processor and 64MB of memory. ActiveBatch will work on a 486 or other processor chips and speeds. The actual hardware requirements vary depending on the job workload you need to perform. In general, ActiveBatch attempts to use memory for greater performance so memory utilization increases with the number of active Queues and Jobs deployed.

Client: A Client-Only installation is supported on Windows 2000, Windows NT Version 4 (running at least Service Pack 4) and Windows 9x/Me on an Intel compatible processor. The actual hardware requirements vary depending on the job workload you need to perform. Individual Client interface requirements are as follows:

- ABATAdmin/COM supports Windows 2000/NT (Intel) and 9x/Me machines
- Command-line interface supports Windows 2000/NT (Intel) and 9x/Me machines.
- MMC Snap-in supports Windows 2000 and Windows NT (Intel)

Windows 95 requires that you have the Windows Common Controls update. This self-extracting executable can be found on the CD in /redist. Windows 98/95 require DCOM support. DCOM 95 is automatically installed as part of the installation. A system reboot may be required due to this object installation.

Execution Agent: An Execution Agent installation is supported on Windows 2000, Windows NT Version 4 (running at least Service Pack 4) (both Intel and Alpha), OpenVMS Version 6.2 (or later version), Compaq Tru64 Version 4 (or later version) and Sun Solaris (Sparc) Version 6 (or later version). The Execution Agent that is installed as part of the standard ActiveBatch installation procedure is applicable to Microsoft Windows operating systems only. The other operating systems have their own installation procedures. Note: RPC or DCE must be installed prior to ActiveBatch installation. Refer to the product Release Notes for platform specific installation instructions and requirements.

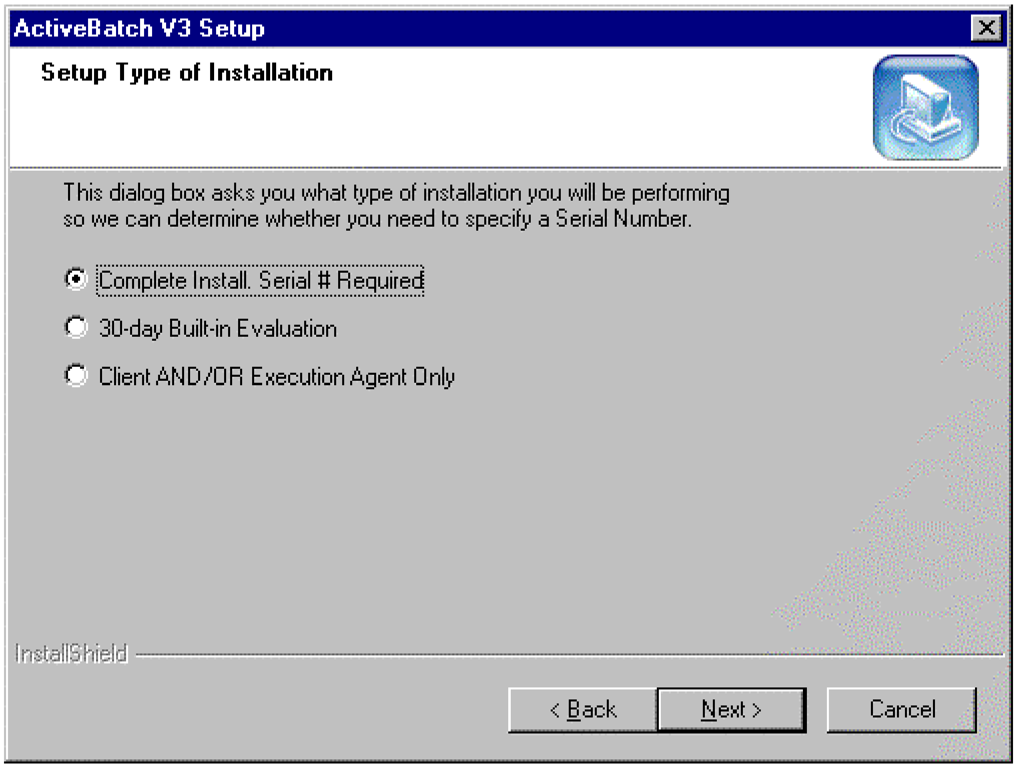
All ActiveBatch components require the use of Microsoft RPC or DCE communication facilities. The ActiveBatch installation will check whether the RPC service has been started and will start the service if necessary. All ActiveBatch services indicate that RPCSS is a required dependency. ActiveBatch services are run using the *LocalSystem* built-in account.

ActiveBatch Client and Job Scheduler components optionally use Microsoft's Windows Management Instrumentation (WMI) software. ActiveBatch will first check for the presence of WMI and then ask whether the installation procedure can install it. While you can deny or defer WMI installation for Windows NT and Windows 9x systems, ASCI highly recommends that you install WMI to reap the maximum benefits of ActiveBatch.

When the client components of ActiveBatch are installed, a check is made to determine whether Microsoft's Management Console (MMC) components are already installed. ASCI requires that MMC V1.2 (or later) be installed, particularly for administrative use, prior to the ActiveBatch installation.

3.1.1 Windows 2000/NT/Me/9x Installation Process

The installation process begins with the *Welcome* screen confirming that you really want to install some or all ActiveBatch components. Clicking on the *Next* button moves forward to the next step in the process. The *Next* button moves you forward, the *Back* button moves you to the preceding dialog. The *Cancel* button allows you to cancel and abort the installation process (after a subsequent confirming message).



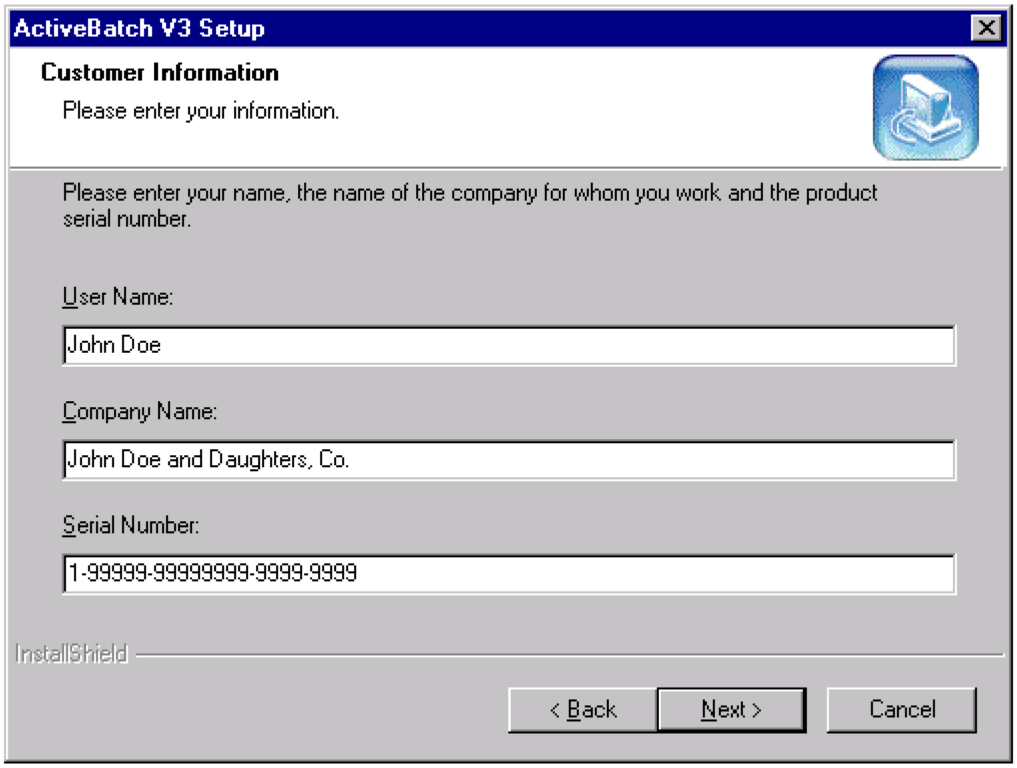
This dialog box asks you what type of installation you will be performing so we can determine whether you need to specify a Serial Number.

- [✓ Complete Install. Serial # Required]
- 30-day Built-in Evaluation
- Client AND/OR Execution Agent Only

Figure 1 - Type of Installation (Windows 2000/NT)

Step 2 requests the type of installation you intend to perform. This information is requested to determine whether a Serial Number key is required. Client and Execution Only installations do not require a Serial Number for installation purposes (Execution Agents do require licensing). The 30-day Built-in Evaluation allows you to evaluate the product without contacting us. After 30-days you will either need to remove the software, request a 30-day Evaluation Serial Number to extend the evaluation or purchase the product and enter your permanent serial number key.

Step 3 is a confirmation of the License Terms and Conditions (as they appear in this manual) that will govern your evaluation or continued use of this product. Please take time to read them carefully. You must agree to the License Terms before continuing. If you don't agree to these terms click the *No* button to stop the installation process. If you do agree to these terms click the *Yes* button to continue.



Please enter your name, the name of the company for whom you work and the product serial number.

User Name: John Doe

Company Name: John Doe and Daughters, Co.

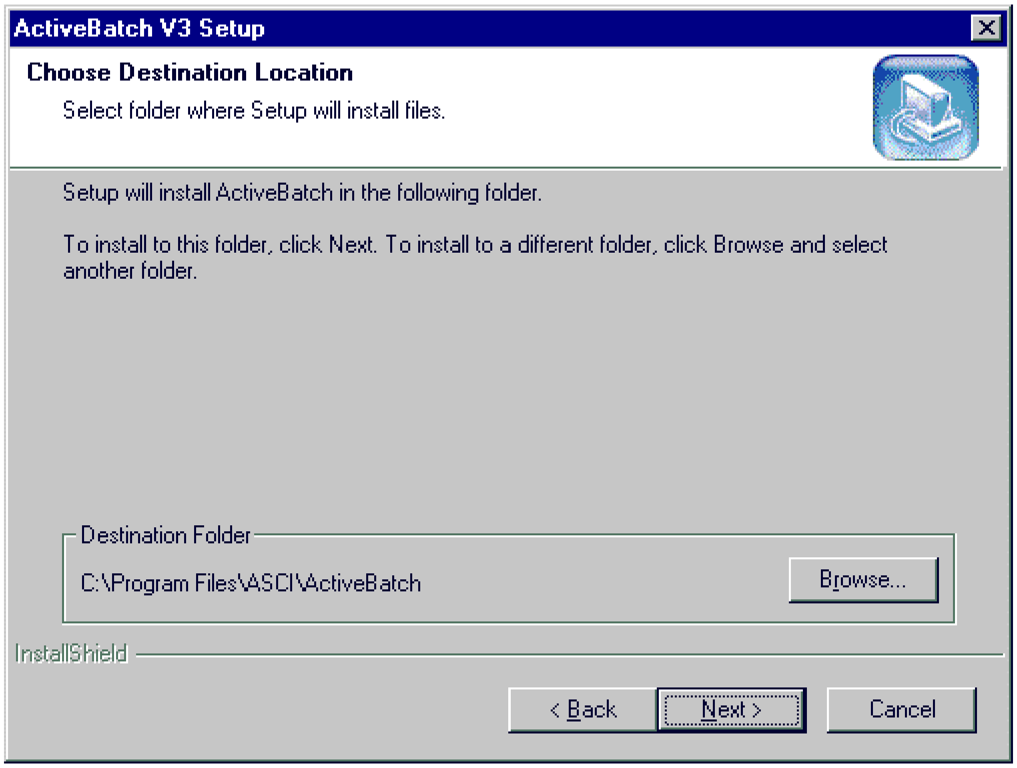
Serial Number: 1-99999-99999999-9999-9999

Figure 2 - Customer Information (Windows 2000/NT/9x/Me)

Step 4 is the Customer Registration Setup screen. This dialog box will display your name and company (if this information has already been provided to Windows). Please enter and/or confirm this information. You need to enter a Serial Number when requesting a Complete or Job Scheduler only installation otherwise no Serial Number field is present.

Once you have entered the required information you can click the *Next* button to proceed. You must then confirm the registration information you entered. Clicking *NO* allows you to correct or re-enter the preceding registration information.

If you requested the 30-day built-in Evaluation you will be reminded that this installation will operate for only thirty (30) days and you must either purchase the software or extend the evaluation period by requesting a new evaluation serial number from ASCI (sales@advsyscon.com) or remove the product from your machine.



ActiveBatch V3 Setup

Choose Destination Location

Select folder where Setup will install files.

Setup will install ActiveBatch in the following folder.

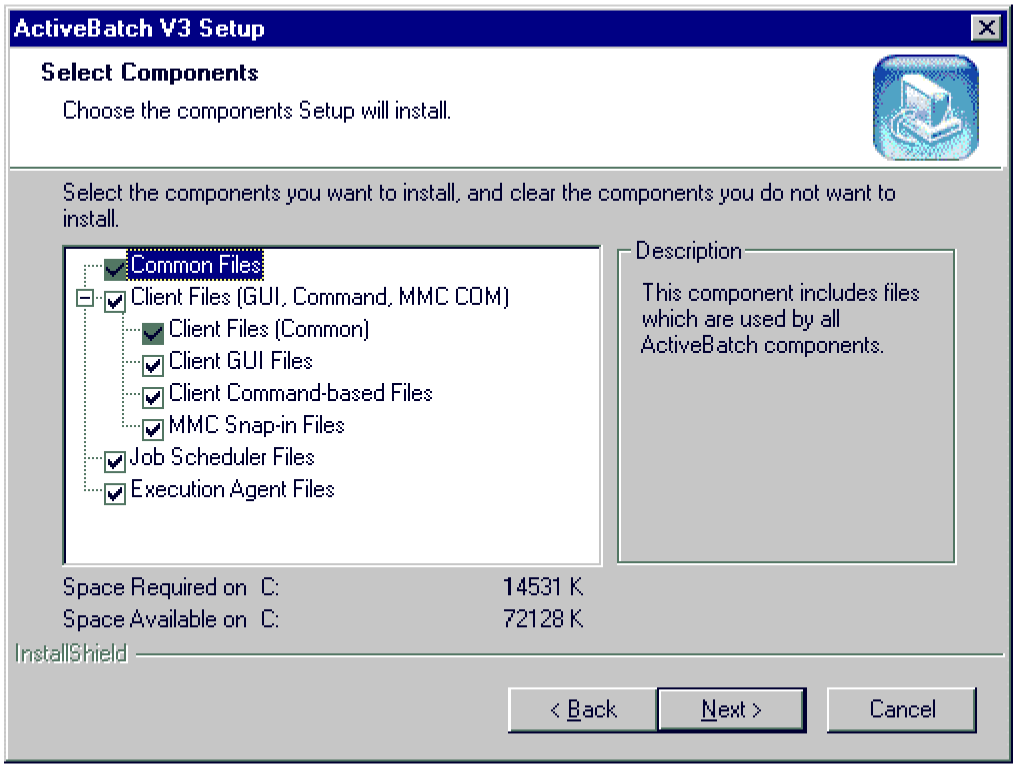
To install to this folder, click Next. To install to a different folder, click Browse and select another folder.

Destination Folder

C:\Program Files\ASCI\ActiveBatch

Figure 3 - Product Location Information (Windows 2000/NT/9x/Me)

Step 5 is the product Destination Location information screen. By default, ActiveBatch will install the product in \Program Files\ASCI\ActiveBatch of the system disk. This folder will be created if it does not already exist. To select a different drive and/or folder name, click on the *Browse* button and select a different location. Click *Next* to continue.


ActiveBatch V3 Setup

Select Components

Choose the components Setup will install.

Select the components you want to install, and clear the components you do not want to install.

- ✓ Common Files
- ✓ Client Files (GUI, Command, MMC COM)
  - ✓ Client Files (Common)
  - ✓ Client GUI Files
  - ✓ Client Command-based Files
  - ✓ MMC Snap-in Files
- ✓ Job Scheduler Files
- ✓ Execution Agent Files

Description

This component includes files which are used by all ActiveBatch components.

Space Required on C: 14531 K

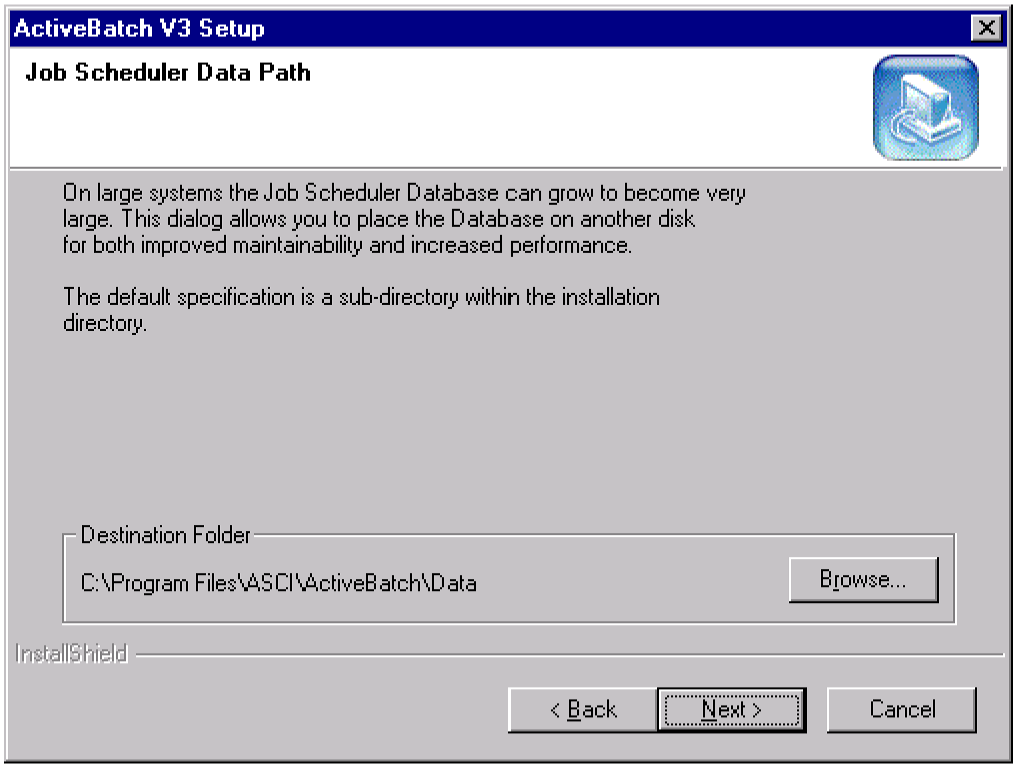
Space Available on C: 72128 K

Figure 4 - Component Selection (Windows 2000/NT/9x/Me)

Step 6 is the Component Selection screen. Components are arranged in tree form so you can visually determine each component and any related sub-components that are needed. A checkmark in a component box indicates that the component has been selected for installation. When a component has a dark colored box that means that component is required and cannot be de-selected. For example, you cannot de-select the *Common Files* component. If you select any or all of the Client components you must also install the *Client Files (Common)* sub-component. With this approach you can completely determine what you want to install. Simply check the checkbox to install the component or clear the checkbox to not install the component's files. If you click on the component name portion, a brief description appears on the right indicating what that component does. The *Space Required on* line indicates the required space the components will take on the disk drive you selected for installation. The second line *Space Available on* provides a handy visual reference for the approximate space on the target drive.

If your machine doesn't have the Microsoft Management Console (MMC) software installed you will be reminded to install it. ActiveBatch's MMC Snap-in has been tested against Microsoft's MMC V1.2 (or later). For CD installations an opportunity is provided to install the MMC software.

If your machine doesn't have the Microsoft Windows Management Instrumentation (WMI) software installed, the installation process will remind you to install it or, for CD installations, allow you an opportunity to execute the WMI installation software.



On large systems the Job Scheduler Database can grow to become very large. This dialog allows you to place the Database on another disk for both improved maintainability and increased performance.

The default specification is a sub-directory within the installation directory.

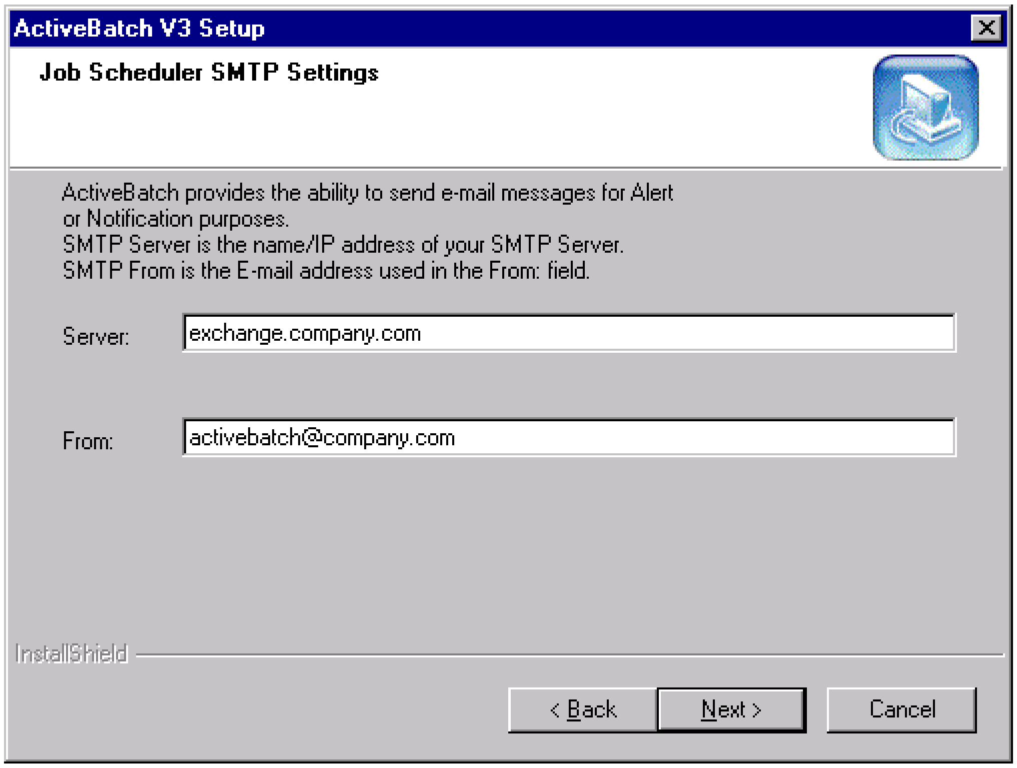
Destination Folder

C:\Program Files\ASCI\ActiveBatch\Data

Figure 5 - Job Scheduler Data Path Dialog (Windows 2000/NT)

Step 7 appears only if you elect to install the ActiveBatch Job Scheduler component.

Depending on the number of batch jobs you intend to execute and the amount of job history you wish to retain, the ActiveBatch database can grow to become quite large. This dialog allows you an easy opportunity to locate the database portion of the product to another disk. You can increase product performance by locating the database files to another non-system disk. By default, the installation will locate the database files to a sub-directory within the original installation path.



ActiveBatch provides the ability to send e-mail messages for Alert or Notification purposes.
SMTP Server is the name/IP address of your SMTP Server.
SMTP From is the E-mail address used in the From: field.

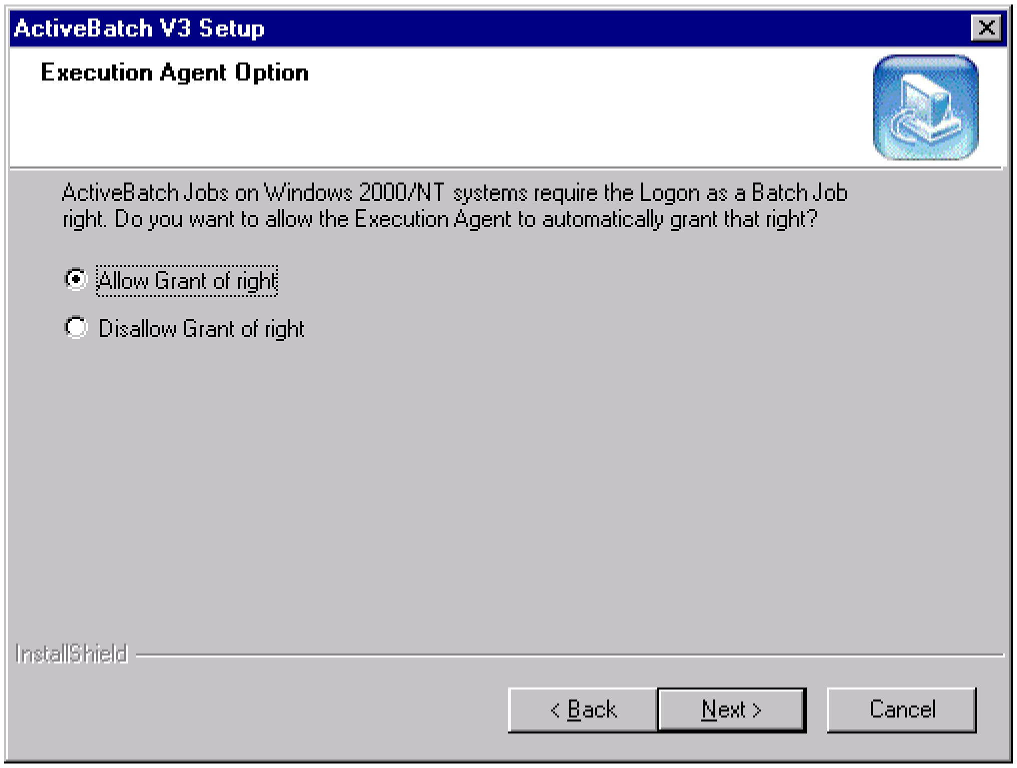
Server: exchange.company.com

From: activebatch@company.com

Figure 6 – Job Scheduler SMTP Settings (Windows 2000/NT)

Step 8 only appears if you elect to install the ActiveBatch Job Scheduler component.

ActiveBatch provides the ability to send SMTP E-mail for alert and notification purposes. SMTP mail requires that ActiveBatch know the fully qualified hostname or IP address of the SMTP server. The “From:” field represents a valid e-mail address on the SMTP server that will be used to identify the source of the mail message.



ActiveBatch Jobs on Windows 2000/NT systems require the Logon as a Batch Job right. Do you want to allow the Execution Agent to automatically grant that right?

[✓ Allow Grant of right]

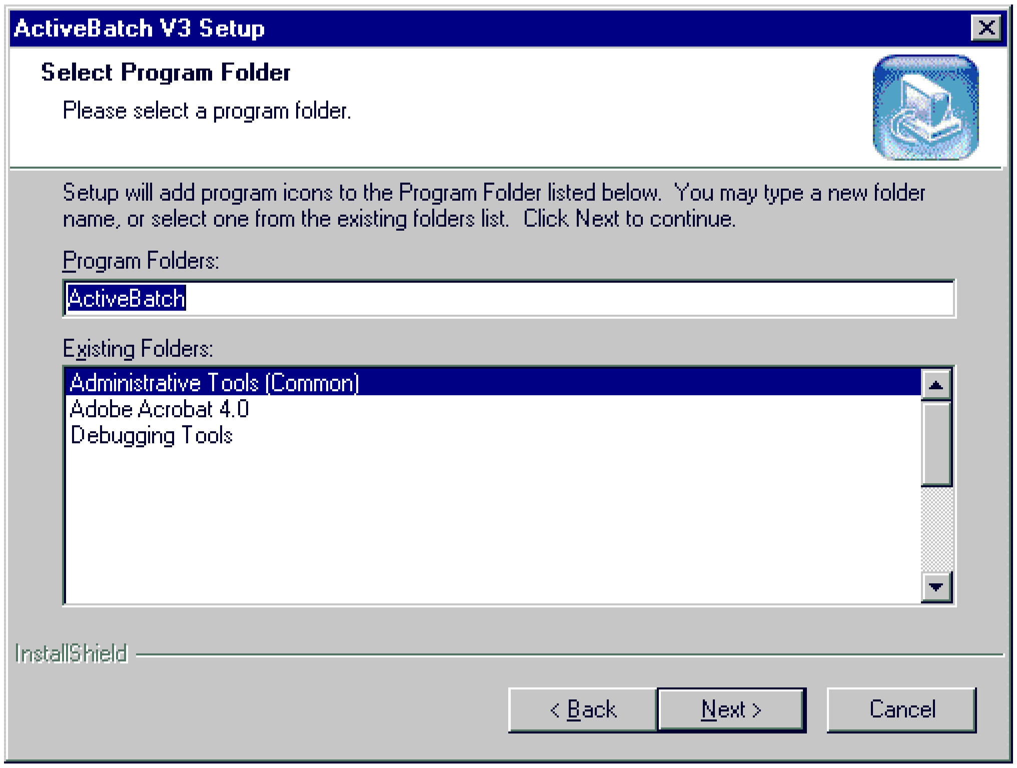
Disallow Grant of right

Figure 7 - Execution Agent Option Dialog (Windows 2000/NT)

Step 9 only appears if you elect to install the Execution Agent component.

The ActiveBatch product requires that Windows 2000/NT users be granted the "Logon as a Batch Job" right. Section 6.5.1 describes how to manually add the right to a user or group account. You can avoid the manual requirement by enabling the automatic grant of this right on demand. By default, the ActiveBatch Execution Agent will transparently grant this local right whenever a batch logon would otherwise fail. When the Execution Agent grants this right an NT Event Log entry is made noting the grant.

If you feel uncomfortable with this feature, click the "Disallow…" radio button. You will then have to manually grant this right to all user/group accounts, for each machine, that you intend to execute jobs on.


ActiveBatch V3 Setup

Select Program Folder

Please select a program folder.

Setup will add program icons to the Program Folder listed below. You may type a new folder name, or select one from the existing folders list. Click Next to continue.

Program Folders:

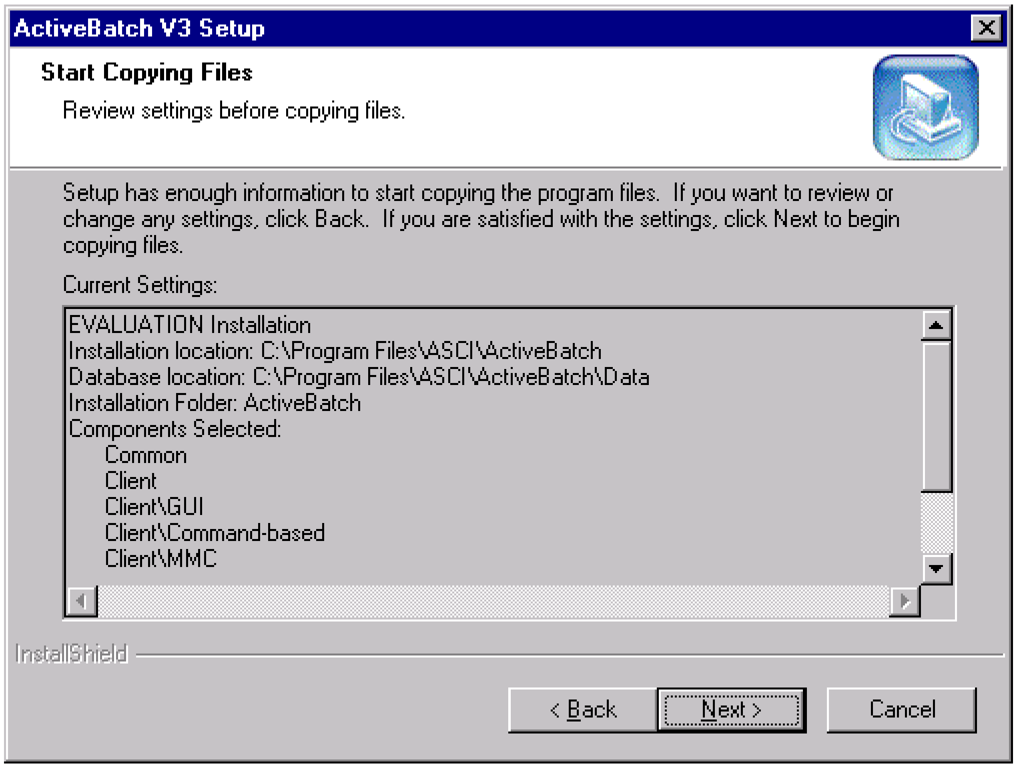
ActiveBatch

Existing Folders:

Administrative Tools (Common)
Adobe Acrobat 4.0
Debugging Tools

Figure 8 - Select Program Folder (Windows 2000/NT/9x/Me)

Step 10 is the Program Folder selection screen. This dialog box allows you to specify the name of the ActiveBatch folder. By default, the name is "ActiveBatch." You may specify a different folder by entering the new name in the area titled "Program Folders" or selecting one from the list of existing folders.



Start Copying Files

Review settings before copying files.

Setup has enough information to start copying the program files. If you want to review or change any settings, click Back. If you are satisfied with the settings, click Next to begin copying files.

Current Settings:

```
EVALUATION Installation
Installation location: C:\Program Files\ASCI\ActiveBatch
Database location: C:\Program Files\ASCI\ActiveBatch\Data
Installation Folder: ActiveBatch
Components Selected:
    Common
    Client
    Client\GUI
    Client\Command-based
    Client\MMC
```

Figure 9 - Confirm Final Settings (Windows 2000/NT/9x/Me)

Step 11 is the Review Settings Confirmation screen. This dialog box will redisplay all the settings you selected. To continue and actually start the installation phase you must click on the *Next* button. If you decide to change any settings simply click on the *Back* button and review any dialog input you made.

If you elect to continue with the installation a progress bar will appear indicating what type of files are being copied and how much longer you have until the installation finishes. Barring error the installation will continue without any user input or additional interaction until the end.

If you installed the Job Scheduler component, the ActiveBatch Scheduling database will be created (unless you already have a database present). The Job Scheduler service will then be started. For Windows 2000, by default, the Job Scheduler service is configured to allow itself to be automatically restarted twice within a twenty-four (24) hour time period. You can change the recovery options of this service using the Windows 2000 Services application.

If you are installing the Execution Agent component a share named ASCI_ABATLOG will be created. This share points to the ActiveBatch sub-directory named *ExecLogFiles*. This sub-directory is used as the root of the ActiveBatch Centrally Managed Log Files feature. The share is used to allow other ActiveBatch users the ability to view their log files. The Execution Agent service will then be started. For Windows 2000, by default, the Execution Agent service is configured to allow itself to be automatically restarted twice within a twenty-four (24) hour time period. You can change the recovery options of this service using the Windows 2000 Services application.

If you are installing the Command Line Client or Execution Agent components, the system PATH variable will be updated with the ActiveBatch installation directory.

Note: If you install WMI after ActiveBatch installation

If you install WMI after you have installed ActiveBatch you will need to issue one simple console command to enable WMI to process ActiveBatch events:

```
MOFCOMP AbatJSSEvent.MOF
```

The above command invokes the WMI MOF compiler and compiles the ActiveBatch MOF text file. *MOFCOMP* is typically located under the Windows System Directory within the WBEM sub-directory. *AbatJSSEvent.MOF* can be found in the ActiveBatch Installation directory.

3.2 Managing ActiveBatch Licenses

Depending on the product edition you purchased, your license may allow you to purchase and deploy additional Execution Agents. An Execution Agent is a portion of the ActiveBatch software responsible for the execution of jobs in the system. The Personal Edition allows for no additional agents. The Small Business Server Edition allows you to purchase up to nine (9) additional agents (all of which are restricted to Microsoft operating systems). The Advanced Server Edition allows you to purchase an unlimited number of additional agents (although the agents are all confined to the Microsoft Windows 2000/NT operating system). The Cross-Platform Enterprise Edition also allows you to purchase an unlimited number of additional agents for all the platforms the product supports.

Execution Agent licenses are purchased and applied to the Job Scheduler system they will be associated with. While an Execution Agent can be shared by more than one Job Scheduler, no license sharing is permitted. In other words, each Execution Agent must be licensed on a per Job Scheduler basis.

Depending on the type of installation of ActiveBatch you performed you may have been prompted for a serial number. While all components of ActiveBatch are licensed, serial number key(s) control what type of Job Scheduler component is used and how many Execution Agents (and what type) are supported. The utility program ABATLIC is used to update ActiveBatch licensing. This program can be used to replace an expired evaluation serial number, replace a product edition serial number and add Execution Agent serial numbers.

ABATLIC is a console application that is installed when the Job Scheduler component is installed and supports five (5) commands: EXTEND, CONVERT, UPGRADE, ADD and LIST.

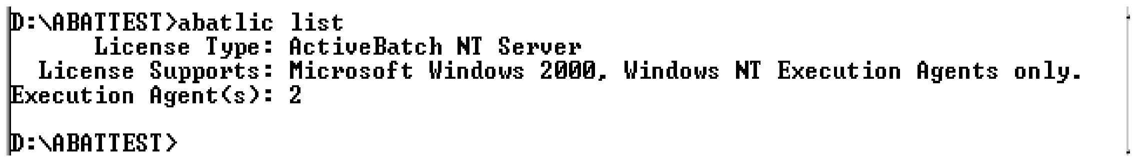
EXTEND allows you to extend an evaluation. You must first obtain a new evaluation serial number from your distributor or ASCI. Of course, you should try and do this before your evaluation expires.

CONVERT allows you to convert an existing evaluation into a permanent installation. You must first purchase and obtain a permanent serial number.

UPGRADE allows you to convert an existing permanent product serial number into a different permanent serial number. You must first purchase and obtain the new serial number.

ADD allows you to add additional Execution Agents. Remember that the ActiveBatch product you installed may have limitations that pertain to the number of supported Execution Agents. This command allows you to add newly purchased Execution Agent license(s).

LIST provides a complete list of all ActiveBatch licensing on your machine.



```
D:\ABATTEST>abatlic list
      License Type: ActiveBatch NT Server
  License Supports: Microsoft Windows 2000, Windows NT Execution Agents only.
Execution Agent(s): 2

D:\ABATTEST>
```

Figure 10 - Working with ABATLIC

3.2.1 ActiveBatch Registry Information

This section describes some of the Registry keys and values that may be modified to alter ActiveBatch component operation. Please note that caution must be exercised whenever the Registry is to be changed. Incorrect Registry alteration may result in an unusable ActiveBatch product or worse an unusable system. ASCI recommends that you use the ActiveBatch MMC Snap-in for this purpose as described in Chapter 0.

All ActiveBatch Registry entries can be found in the HKEY_LOCAL_MACHINE hive under Software\ASCI\ActiveBatch\CurrentVersion. While you will find more registry keys and value names then are described here you should not attempt to modify any unless specifically told to do so by ASCI Technical Support personnel.

The first section of registry items pertain to the Job Scheduler. In the interest of formatting the file specification "\Program Files\ActiveBatch" has been abbreviated to "ActiveBatch\". Of course, the actual installation directory is decided by you when you install the product. After you change any of the registry entries you should invoke the *Parameter Change* feature within the MMC Snap-in.

CopyFolder REG_SZ "ActiveBatch\CopyFiles"

This value name controls the location where jobs are saved when the "Copy File…" feature is enabled as part of a job definition. By default, these files are placed in a sub-directory "CopyFiles" under the main ActiveBatch installation directory. For very heavy usage of this feature you may want to relocate this directory to another disk for better I/O utilization and performance. To relocate these files, move the entire sub-directory to another directory you have created on the target disk. You must then change this registry entry to reflect the new location.

DataPath REG_SZ "ActiveBatch\Data"

This value name controls the location of the database files for ActiveBatch Job Scheduling. By default, these files are placed in a sub-directory "Data" under the main ActiveBatch installation directory. For very heavy Job Scheduling usage you may want to relocate this directory to another disk for better I/O utilization and performance. To relocate the database files, move the entire sub-directory to another directory you have created on the target disk. You must then change this registry entry to reflect the new location. The Job Scheduler must be stopped prior to relocating any files.

SMTPFrom REG_SZ ""

This value name controls the "From:" field on an SMTP E-mail that can be sent by the Job Scheduler for notification or alerts. You should specify a valid e-mail address (e.g. name@commerce.com) that represents a real e-mail destination on your company's e-mail system.

SMTPPort REG_DWORD 0x19

This value name controls the TCP port number for communication to the SMTP server denoted in "SMTPServer". The port number value above is the default port number (25 in decimal) for SMTP server operation.

SMTPServer REG_SZ ""

This value name represents the name of the machine on which an SMTP Server resides. The value should be represented in TCP format. Either as a fully qualified machine domain name (e.g. machine.commerce.com) or an IP address.

The following registry items are for the Windows 2000/NT Execution Agent:

ExecAutoRight REG_SZ "TRUE"

This value name controls whether the Execution Agent may automatically grant the "Logon as a Batch Job" right to users who would otherwise fail to execute their batch jobs on the selected machine. A setting of TRUE enables the feature and a setting of FALSE disables it.

ExecCopyFolder REG_SZ "ActiveBatch\ExecCopyFiles"

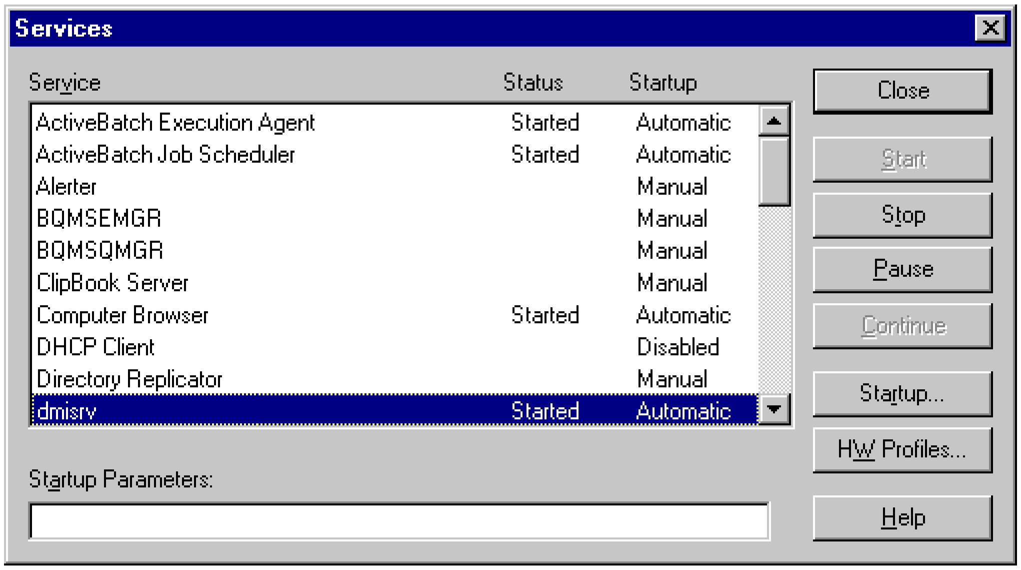
This value name controls the location where jobs are saved on the Execution Machine when the "Copy File…" feature is enabled as part of a job definition. By default, these files are placed in a sub-directory "ExecCopyFiles" under the main ActiveBatch installation directory. For very heavy usage of this feature you may want to relocate this directory to another disk for better I/O utilization and performance. To relocate these files, move the entire sub-directory to another directory you have created on the target disk. You must then change this registry entry to reflect the new location. The Execution Agent should be stopped prior to changing this entry.

ExecLogFolder REG_SZ "ActiveBatch\ExecLogFiles"

This value name controls the location where a series of sub-directories are used for centralized management of ActiveBatch jobs that use the Centrally Managed Log files feature.

3.3 Starting ActiveBatch

We previously discussed the three (3) types of components that make up ActiveBatch. Before you can start ActiveBatch you must have previously licensed and installed the complete software or at least the Job Scheduler on a Windows 2000/NT system. When we speak of "starting" ActiveBatch, we refer specifically to starting the Job Scheduler portion of the product. Most companies will have at least one machine that will act as the Job Scheduler and that machine will be accessed by one or more client machines. The Job Scheduler component is implemented as an NT Service ("AbatJSS") and is installed as part of the product installation process. The Service is typically configured to start automatically when the machine is booted and uses the *LocalSystem* built-in account.



Service	Status	Startup
ActiveBatch Execution Agent	Started	Automatic
ActiveBatch Job Scheduler	Started	Automatic
Alerter		Manual
BQMSEMGR		Manual
BQMSQMGR		Manual
ClipBook Server		Manual
Computer Browser	Started	Automatic
DHCP Client		Disabled
Directory Replicator		Manual
dmisrv	Started	Automatic

Figure 11 - Job Scheduler/Execution Agent NT Services

To actually run jobs on machines, you must have previously licensed and installed the complete software or at least the Execution Agent on the selected machines. On Windows 2000/NT, the Execution Agent is also implemented as an NT Service ("AbatEAgent") and this service is also typically configured to startup when the machine is booted. On non-Microsoft based machines, the Execution Agent is started as part of the script(s) associated with system startup. The common ingredient is that the Execution Agent must be started in order for the Job Scheduler to schedule jobs for it to execute. Once the Job Scheduler and at least one Execution Agent have been started you are ready to use the product through the user interfaces.

3.3.1 Job Scheduler Service Security

By default, the Job Scheduler and Execution Agent run under the *LocalSystem* built-in account. This grants the services whatever local rights are needed for proper operation of the product. "LocalSystem" as the name implies means that this account can only access local and not network objects. Generally, this is not a problem, however, there is one feature of ActiveBatch that you will be precluded from using when operating under this account. By default, if you desire to have ActiveBatch use the Windows NT Event Log for various queue and job logging, the events are sent to the local Job Scheduler's NT Application Event Log. ActiveBatch does allow you to specify another machine for Event Logging. If you would like to use this feature you will need to change the Job Scheduler's service account from LocalSystem to "Administrator" (or an account that is within the Administrators group).

3.3.2 Job Scheduler E-Mail Facility

ActiveBatch provides the ability to send an e-mail as part of a Job Notification and/or Alert request. ActiveBatch's Job Scheduler provides this facility through the use of SMTP. If you would like to use the e-mail facility then you must specify values for the following Registry items:

SMTPFrom **REG_SZ** **""**

This value name controls the "From:" field on an SMTP E-mail that can be sent by the Job Scheduler for notification or alerts. You should specify a valid e-mail address (e.g. name@commerce.com) that represents a real e-mail destination at your company's e-mail system.

SMTPPort **REG_DWORD** **0x19**

This value name controls the TCP port number for communication to the SMTP server denoted in "SMTPServer". The port number value above is the default port number for SMTP server operation.

SMTPServer **REG_SZ** **""**

This value name represents the name of the machine in which an SMTP Server resides. The value should be represented in TCP format. Either has a fully qualified machine domain name (e.g. machine.commerce.com) or an IP address.

SMTPServer and SMTPFrom are the most important as these values indicate the SMTP server that is to be used for e-mail and who the e-mail is to be considered from.

3.4 Stopping ActiveBatch Services

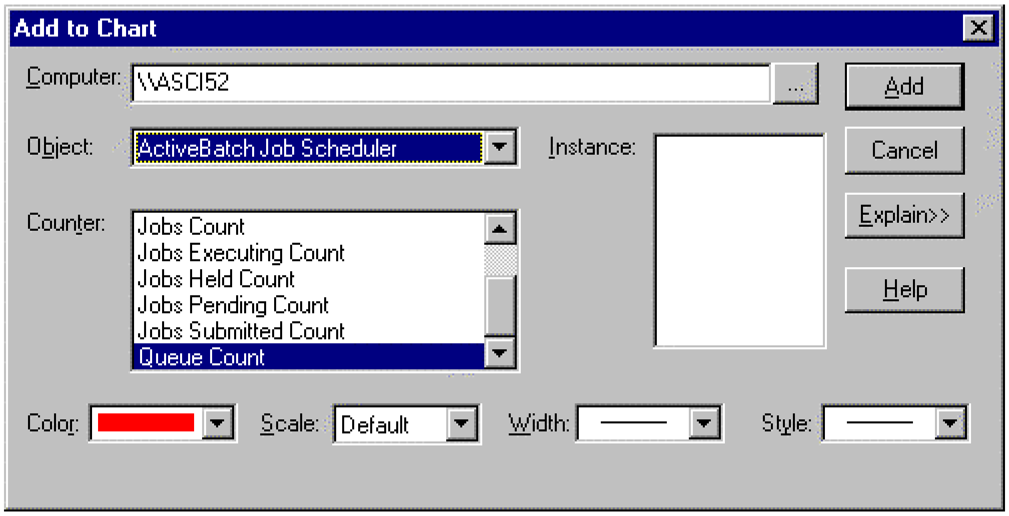
To stop the ActiveBatch Job Scheduler, use the MMC Snap-in or Control Panel Services applet, and look for the Service named "ActiveBatch Job Scheduler". Obviously, you should attempt to stop the Job Scheduler when no ActiveBatch jobs are running or other work is being performed. That said, stopping the Job Scheduler will not impact any jobs that are actually executing since the Execution Agents are actually running the job(s).

The best approach to stopping an Execution Agent is to first stop all execution queues that are being directed to that agent. This way any outstanding jobs have an opportunity to complete and new jobs won't be dispatched to the agent you intend on stopping. To stop the ActiveBatch Execution Agent, use the MMC Snap-in or Control Panel Services applet, and look for the Service named "ActiveBatch Execution Agent".

3.5 Monitoring ActiveBatch Performance

Once ActiveBatch is installed and started you can monitor an ActiveBatch Job Scheduler machine from any Windows 2000 or NT system through the Performance Monitor facility. This facility is built into every Windows 2000 or NT machine and can be found in the Administrative Tools section.

ActiveBatch provides two (2) objects whose metrics can be monitored using the Performance Monitor: Job Scheduler and Queue. Both objects are actually preceded by the word “ActiveBatch”. Clicking the “Add to Chart” (under Edit) or the “plus” sign (on the Tool bar) causes the following dialog box to appear. Click on the Object list to select one of the above objects.



Computer: \\ASCI52

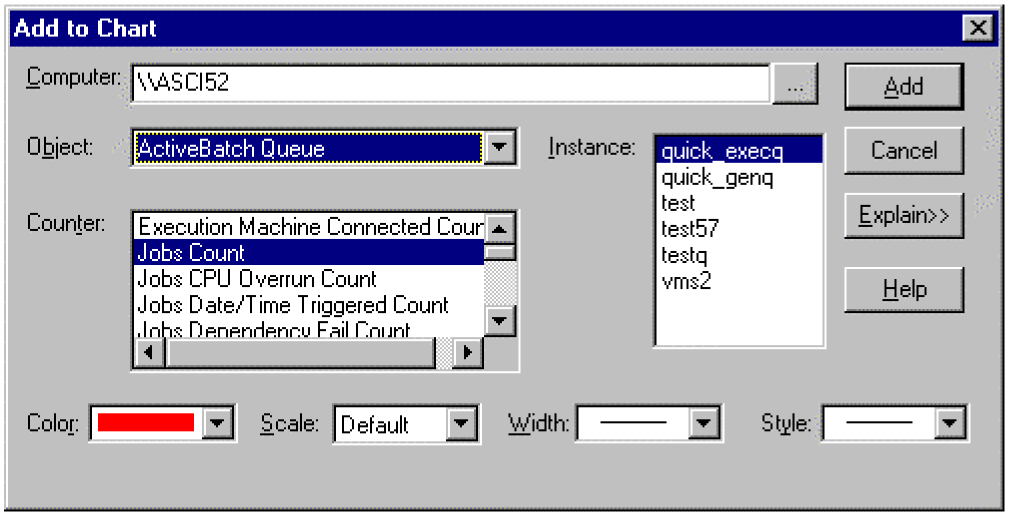
Object: ActiveBatch Job Scheduler

Counter:
Jobs Count
Jobs Executing Count
Jobs Held Count
Jobs Pending Count
Jobs Submitted Count
Queue Count

Color: 🟥 Scale: Default Width: Style:

Figure 12 - Add to Chart "Job Scheduler"

A series of counters are displayed for the object. Counters are either *Operational* or *Historical*. An Operational counter is one in which the value is retained in a non-volatile manner and is true regardless of Job Scheduler startups. An Historical counter is one in which the value is volatile and is reset to zero each time the Job Scheduler is started. For example, *Jobs Executing Count* is operational while *Jobs Submitted Count* is historical. You’ll note that the Job Scheduler object has no instances because there is only one Job Scheduler allowed per machine. However the second object, ActiveBatch Queue will usually contain multiple instances – one for each Queue on the machine. Figure 13 illustrates this with several queues listed as “Instances”.



Computer: \\ASCI52

Object: ActiveBatch Queue

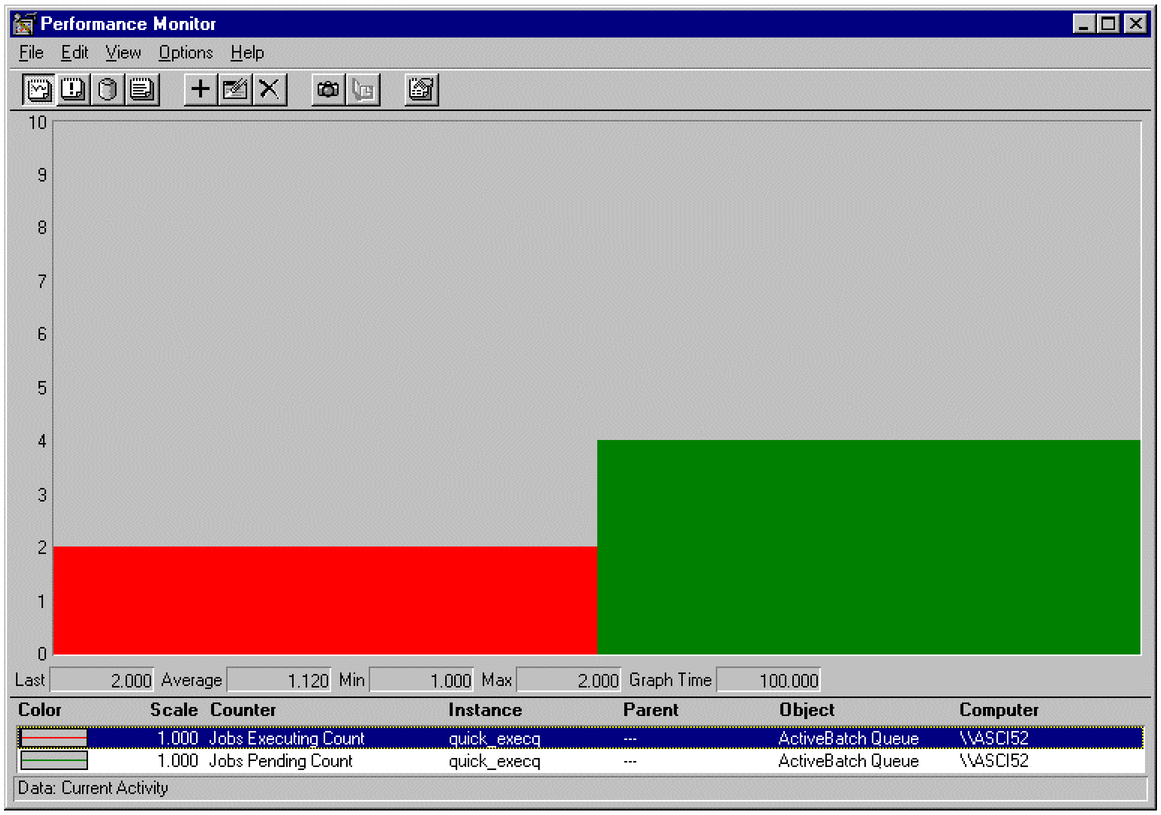
Instance: quick_execq, quick_genq, test, test57, testq, vms2

Counter: Execution Machine Connected Coun, Jobs Count, Jobs CPU Overrun Count, Jobs Date/Time Triggered Count, Jobs Dependency Fail Count

Color: 🟥 Scale: Default Width: Style:

Figure 13 - Add to Chart "Queue"

You can select a counter for each instance and you can always add multiple object/instance/counters for display.



Last	Average	Min	Max	Graph Time
2.000	1.120	1.000	2.000	100.000

Color	Scale	Counter	Instance	Parent	Object	Computer
🟥	1.000	Jobs Executing Count	quick_execq	---	ActiveBatch Queue	\\ASCI52
🟩	1.000	Jobs Pending Count	quick_execq	---	ActiveBatch Queue	\\ASCI52

Data: Current Activity

Figure 14 - ActiveBatch Performance Monitoring

For example, the above figure displays a histogram for the Pending and Executing Job counters for queue “quick_execq”. The left column is *Executing* and the right column is *Pending*. These counters are also available for WMI usage through the WMI supplied Performance Monitor Provider.

3.6 ActiveBatch DBPurge

The ActiveBatch database consists of pending, active and completed jobs. A completed job contains all aspects and details about the job as well as any log file(s) that were written when the job executed. As you can imagine this can be a lot of data to keep and, while important, for most customers once a job completes successfully much of the job's history is unwanted. In ActiveBatch the component that manages and prunes unnecessary job history away is DBPurge. DBPurge is run within the ActiveBatch Job Scheduler each night and prunes away job history records that are no longer to be retained as well as a job's log files. This later deletion occurs only if the log file(s) were centrally managed. DBPurge obeys the retention criteria established for Job History and Log File. You may also run DBPurge manually through the ActiveBatch MMC Snap-in. DBPurge requires that the Job Scheduler be started.

By default, the Job Scheduler runs the DBPurge program every morning at 12:00am (midnight). You can change that default through the System Policies "DBPurgeRun" and "DBPurgeTime". Read Section 8.4 for more information.

By default, DBPurge will also produce a log of all its activities. The log file can be found at <ActiveBatch Install Path>\DBPurge.LOG. The last seven (7) runs performed are retained and the logs are not cumulative.

3.7 Migrating from BQMS

ActiveBatch represents a major new release in terms of previous functionality provided in BQMS Version 2. The good news is that BQMS features are present within ActiveBatch and BQMS client commands are generally upward compatible (with some notations). This section discusses the migration strategy that current BQMS customers should use to migrate to ActiveBatch.

First, ActiveBatch is not BQMS. While many commands look similar the products are actually quite different. ActiveBatch is a brand new design and “from scratch” implementation. Next, XLNT is no longer packaged with ActiveBatch or BQMS. XLNT V3 (or later) does provide support for ActiveBatch commands, however, the other portions of ActiveBatch are no longer packaged with XLNT. The last XLNT release to support BQMS is Version 2. The first XLNT release to support ActiveBatch is Version 3. Does this mean you need to buy XLNT Version 3 for ActiveBatch command line support? NO. ActiveBatch comes with built-in command-line support through the ABAT command. If you accessed BQMS command-line facilities through XLNT directly, you will need to upgrade to XLNT Version 3 OR convert your BQMS commands to use the ABAT command prefix (see Section 0). One of the areas of feedback we received from many customers was the use of the XLNT command prefix when accessing BQMS commands from CMD. That is the primary reason we decided to create a new ABAT command. Your continued use of the XLNT command prefix is supported as long as the Windows PATH environmental variable looks to resolve the XLNT.EXE program from the ActiveBatch installation directory and not from the XLNT/BQMS installation directory. This is especially important if you continue to use XLNT Version 2 on the same machine as ActiveBatch. Once you convert to XLNT Version 3, this PATH restriction is not applicable.

3.7.1 BQMS Command Compatibility

ActiveBatch supports most of the BQMS command set. This section discusses those minor qualifiers or features that have no counterpart in ActiveBatch.

1. The /EVERY qualifier.

   ActiveBatch does support the /EVERY qualifier (with some restrictions). This qualifier was used to establish a recurring job using time specifications. Now that ActiveBatch supports the ability to create formal Schedules (which are also much more powerful than /EVERY) we highly recommend that you ultimately convert any /EVERY batch jobs to use the new Schedules feature. ActiveBatch internally converts a SUBMIT/EVERY into a private Schedule anyway and thus the /EVERY qualifier is quite slow in comparison to direct use of Schedules.

2. Retain.

   BQMS provided the ability to retain a job on queue depending on whether the job failed or simply completed. The purpose of "retain" was to visually view the job and job details in the event of an error. ActiveBatch now provides a Job History and retention facility which is much more powerful than "retain" with the added benefit of the job no longer retained on a queue. ActiveBatch performs the following processing when a /RETAIN qualifier appears. First, if /NORETAIN is specified (the default), job history retention is set to one (1) day. /RETAIN=ALL and /RETAIN=ERROR set job history retention to two (2) days. The RETAIN qualifier that could be specified on a Queue is ignored.

3.8 Non-Microsoft Execution Agents

Execution Agents for other operating systems have their own installation instructions appropriate to their platform. The sections that follow provide some of the installation, post-installation and startup instructions for several non-Microsoft platforms. Please check our web site at http://www.activebatch.com for the latest information on newly added Execution Agent platforms.

3.8.1 Compaq TRU64 Unix

3.8.1.1 Installation

The installation software is provided via a TAR file. Once you restore the contents of the TAR file back to disk, enter the following command to start the installation:

```
#./abatinstall
```

The installation procedure also provides a series of *man* pages which can be referenced on-line.

3.8.1.2 DCE/RPC Configuration

The ActiveBatch Execution Agent for TRU64 UNIX requires DCE/RPC V3.1 or later. The DCE/RPC client services at least needs to setup and running correctly for the Execution Agent to start. To configure DCE/RPC on TRU64 UNIX, execute the following command:

```
#dcesetup
```

This will start a menu-driven program that will let you configure DCE/RPC services.

3.8.1.3 Execution Agent Configuration

Configuration of the Execution Agent can be easily performed by invoking the ‘abatconfig’ script. This script can also be used to start the Execution Agent as part of your system startup.

3.8.1.4 Startup and Shutdown

The Execution Agent runs as a daemon and can be run only by root. abatemgr supports the following switches:

```
abatemgr [-h | -v | -s | -t ]
```

-h	provides a simple help message
-v	displays the version number of the software
-s	displays the current status of the Execution Agent

-t terminates the Execution Agent

3.8.1.5 Batch Job Environment and Considerations

Batch Jobs are created as forked processes on TRU64 UNIX and are run under the context of the user whose username and password are supplied as part of the Job properties.

The job specified in the Job Filename field of the Job properties must have the execute permission set and accessible by the user under whom the Job is to be run.

The default working directory for Jobs where the user does not specify a working directory is set to the user's home directory.

If the specified Job to be run is a UNIX shell script, the script's first line must have the standard shell prefix (e.g. #!/bin/sh) specified for it to run correctly.

3.8.2 Sun Solaris

3.8.2.1 Installation

The installation software is provided via a TAR file. Once you restore the contents of the TAR file back to disk, enter the following command to start the installation:

```
#./abatinstall
```

The installation procedure also provides a series of *man* pages which can be referenced on-line.

3.8.2.2 DCE/RPC Configuration

The ActiveBatch Execution Agent for Sun Solaris requires DCE Base Services from IBM. The DCE/RPC client services needs to be setup and running correctly for the Execution Agent to start. To configure DCE/RPC on Sun Solaris, perform the following steps:

1) Execute the following command to configure DCE:

```
# config.dce -config_type local -cell_name cell all_cl
```

2) Confirm installation of DCE/RPC services by executing "show.cfg".

3.8.2.3 Execution Agent Configuration

Configuration of the Execution Agent can be easily performed by invoking the ‘abatconfig’ script. This script can also be used to start the Execution Agent as part of your system startup.

3.8.2.4 Startup and Shutdown

The Execution Agent runs as a daemon and can be run only by root. abatemgr supports the following switches:

```
abatemgr [-h | -v | -s | -t ]
```

-h	provides a simple help message
-v	displays the version number of the software
-s	displays the current status of the Execution Agent
-t	terminates the Execution Agent

3.8.2.5 Batch Job Environment and Considerations

Batch Jobs are created as forked processes on Sun Solaris and are run under the context of the user whose username and password are supplied as part of the Job properties.

The job specified in the Job Filename field of the Job properties must have the execute permission set and accessible by the user under whom the Job is to be run.

The default working directory for Jobs where the user does not specify a working directory is set to the user's home directory.

If the specified Job to be run is a UNIX shell script, the script's first line must have the standard shell prefix (e.g. #!/bin/sh) specified for it to run correctly.

3.8.3 Compaq OpenVMS

3.8.3.1 Installation

Installation of the ActiveBatch Execution Agent for OpenVMS is performed using the VMSINSTAL procedure. No LMF PAK is required since all licensing is performed on the Job Scheduler machine. The saveset name is ABATOVMS030.A. If you've received this file from a web site or via FTP you will need to issue the following VMS command to restore the file's attributes:

```
$ SET FILE/ATTTRIB=(ORG:SEQ,RFM:FIX,LRK=32256) ABATOVMS030.A
```

Then you can invoke @SYS$UPDATE:VMSINSTAL to install the product.

3.8.3.2 DCE/RPC Configuration

Prior to starting the Execution Agent you must first start one of the following TCP/IP stacks: Compaq TCP/IP services, Process Software MultiNet or TCPware. Then you need to start DCE in at least the RPC only mode. With OpenVMS V7.1 (or later) systems, you can install the DCE Run-time Services at no additional charge (no special LMF PAK is required – the software is licensed through the VMS PAK).

Configuring DCE in RPC Only mode is pretty simple. Invoke @SYS$STARTUP:DCE$SETUP and select the "Configure" menu option and "RPC Only" sub-option. The only prompt is the hostname of the machine (which is already specified as the default). Once the configuration is performed, the process DCE$DCED will be created. Subsequent startups are performed through @SYS$STARTUP:DCE$RPC_STARTUP which should be inserted into your site-specific startup procedure after the TCP/IP network software has been started.

3.8.3.3 Execution Agent Configuration

The file ABATOVMS_PRODUCT:AbatEmgr.Conf is a standard ASCII file that contains various parameters that can affect the Execution Agent's run-time characteristics. The only parameter you may change is CPUSpeed. This parameter is used by generic queues to help determine workload balancing.

3.8.3.4 Startup and Shutdown

To start the Execution Agent, invoke @SYS$STARTUP:ABATOVMS_STARTUP. You will normally place this file into your site-specific startup procedure after the TCP/IP and DCE/RPC software has been started.

3.8.3.5 Batch Job Environment and Considerations

The Batch job is started as a detached process. Two types of batch file extensions are supported: .COM and .EXE. All file specifications entered must adhere to VMS specifications. This means you can also specify logical names or any other constructions that would be supported on a VMS system. The ActiveBatch environment variables are created as job-wide logical names. All batch jobs are executed under the security context as specified in the Job *Execution* tab. Please note that the OpenVMS lexical F$MODE will return “INTERACTIVE” and not “BATCH” since this process is not running under the VMS Batch Symbiont.

To use the ABATCHKPT utility we recommend you create the following symbol as part of your LOGIN or SYLOGIN procedure:

```
$ ABATCHKPT :== $ABATOVMS_PRODUCT:ABATCHKPT.EXE
```

4 ACTIVEBATCH TERMS AND CONCEPTS

ActiveBatch is a Job Management and Scheduling System. The *Batch* portion of ActiveBatch refers to the concept of submitting a script or program for non-interactive processing. Simply stated, this means that you don't have to log into Windows NT or remain logged in for your non-interactive script to execute. When a script or program is submitted to ActiveBatch for processing that script or program is said to become a *Batch Job*. Batch jobs allow for work to be scheduled and the results of the job to be analyzed. Batch jobs also offer the potential of utilizing your machines during their *off-hours* when interactive usage is presumably at its lowest.

4.1 Architecture

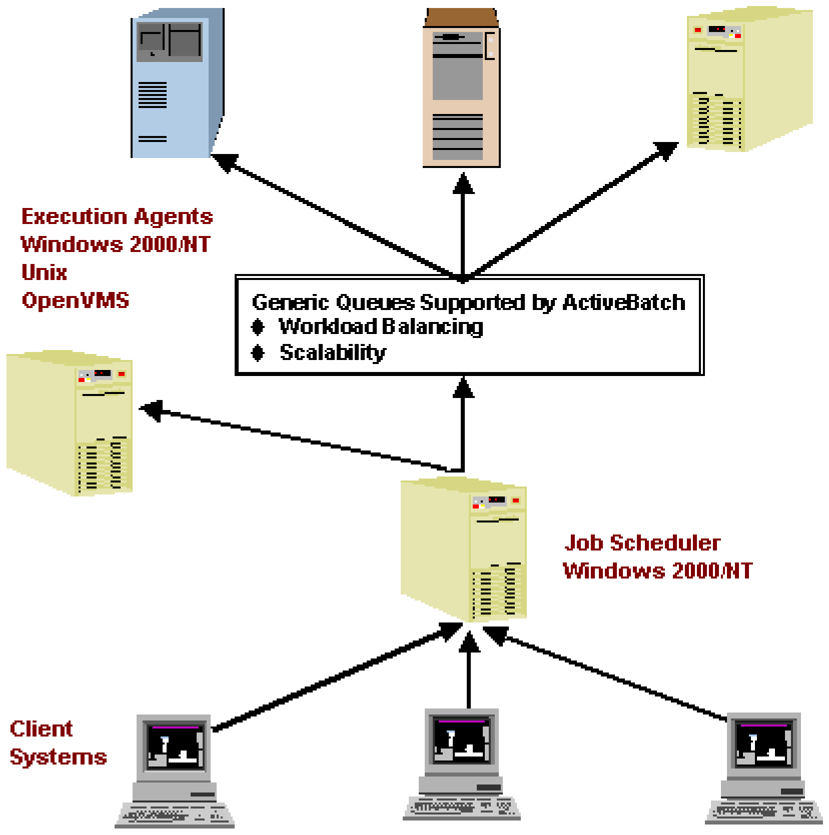


Figure 15 - ActiveBatch Architecture

ActiveBatch is entirely client-server based and all aspects of ActiveBatch can be fully distributed. ActiveBatch consists of three major components: the ActiveBatch **UI clients**, the ActiveBatch **Job Scheduler** and the ActiveBatch **Execution Agent**. All of these components could reside on a single machine or on multiple machines and each component can have multiple instances.

The ActiveBatch UI is the user-interface portion of the ActiveBatch system and provides the ability for users to command various aspects of the product. Three (3) UI components are provided. All UI components communicate with one or more selected Job Scheduler(s). The ABATAdmin program is a standard Windows GUI program that provides a windows interface for working with ActiveBatch. The ABAT program is a command-line based user interface that allows access to the product in either an interactive or non-interactive fashion. System Administrators also have access to the ActiveBatch MMC snap-in. This snap-in provides administrators with the ability to regulate ActiveBatch system usage. The ActiveBatch UI clients communicate with the ActiveBatch Job Scheduler component.

4.2 Objects

The ActiveBatch Job Scheduler is responsible for the maintenance and overall operation of the ActiveBatch system and its databases. The Job Scheduler manages several types of objects the most important of which are Jobs and Queues.

4.2.1 Job Object

A **Job** represents a script or program that is to be executed in a non-interactive[2] manner. Jobs are run under the security context of the submitting user (by default, unless different authentication credentials are provided). A script file is normally specified as the main job step. The script itself can be written using any scripting language installed on the execution machine. The only requirement is that the script extension be registered on the Execution Machine and associated with an actual program or system that is supporting the script. Jobs are submitted for eventual execution and can be triggered for execution in a variety of ways. A *trigger* is an action that causes a job to begin execution. The most typical trigger is date and time and the most typical date and time trigger is "immediate" execution. In the ActiveBatch system, jobs are submitted or queued for execution by specifying a named **Queue**.

Since Jobs can be executed in a recurring or non-recurring fashion, several new terms must be introduced. An **instance** is the actual executing job. A **single instance** or **normal** job is a non-recurring job. This type of job runs and when it completes it's essentially done. A **template** job is a recurring job. This type of job runs on either a schedule or is triggered through events or other aspects of the ActiveBatch system. A template job will cause an **instance** of the job to be created when a trigger is activated (for example, when the scheduled execution time is met). A template job can support one or more instances of the job. Some operations require that you confirm with ActiveBatch whether the operation is intended for the template and/or the instance.

4.2.2 Queue Object

In ActiveBatch a queue represents a non-volatile priority ordered securable object container in which jobs reside waiting for execution. Two types of user defined queues are available: **Execution** and **Generic**. A third type of queue, which is built into ActiveBatch, is known as

[2] ActiveBatch jobs are executed non-interactively from the point-of-view that the system's desktop or console need not be logged on. The term "non-interactive" is not meant to imply that an ActiveBatch job cannot query or execute applications that may be interactive in nature.

OnDemand.

An Execution Queue is a named object in which jobs wait for execution for a specific machine. ActiveBatch allows you to create multiple execution queues and associate each execution queue with a given machine. You can also create multiple execution queues for a single machine. When a batch job is ready to execute, the ActiveBatch Execution Agent, residing on the named machine, is responsible for running the job.

A Generic Queue is essentially a "virtual" machine. A Generic Queue represents one or more Execution Queues. This abstraction is very useful in that the actual machine selected to run the job can be determined dynamically versus the static assignment used when submitting a job to a specific Execution Queue. When a job is ready to be executed, characteristics specified with the job are compared to characteristics specified with the Execution Queues. In addition, Generic Queues can employ "scheduling algorithms" to further refine where the job is to be run.

The OnDemand Queue is a special ActiveBatch built-in queue that is not a Generic queue or associated with any execution machine. Normally, a job is queued to a Generic or Execution Queue when the job is actually defined and submitted to the system. The purpose of the OnDemand queue is to allow a job author to place a job into the ActiveBatch system and defer where the job is to actually execute until a later time. This provides some very interesting benefits which we will cover in Section 6.6.

4.2.3 Schedule/Calendar Objects

A Schedule Object represents some recurring date and time trigger. One or more schedules can be associated with a Job and one or more jobs can share a schedule. Schedules can be public or private. A private schedule can only be associated with one job and when the job's template is deleted so is the schedule. A Schedule is a securable entity and you can assign permissions to allow some, all or no sharing of triggers. ActiveBatch does support the concept of "Business Day." A Business Day is a day when business will be conducted. To determine a business day requires that the Job be assigned to a *Calendar*. A Calendar contains rules for determining holidays and non-working days. When a Job is assigned to a Calendar, the Calendar "filters" any Schedule triggers and ensures that the job can only execute on a "business day" (the Trigger command can override this). The advantage of filtering triggers with a calendar is that the job author need not create special "business" schedules. Also, an administrator can still execute administrative jobs, which are typically scheduled without regard to business days, and use existing schedules. The job, in this case, would run without an assigned calendar. A Calendar is also a securable entity.

4.2.4 Event Object

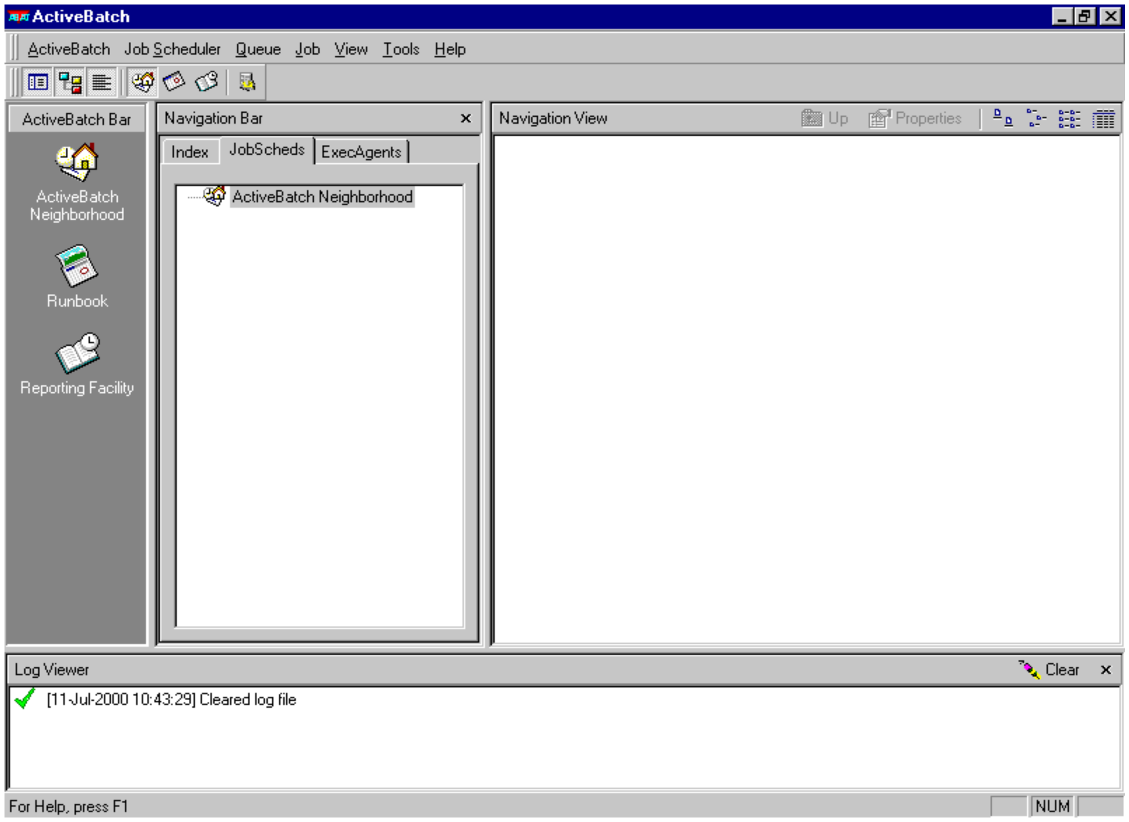
An Entity Object represents one or more Microsoft Windows Management Instrumentation (WMI) events. WMI is used by Microsoft Windows 2000 (built-in) and Windows NT V4 SP4 and/or Windows 9x/Me to centrally manage data from a disparate number of sources. One of the major features of WMI is its event processing. Rather than “polling” to determine whether a job should actually run or not, a WMI event query can be initiated to indicate whether a series of actions have occurred. So rather than run a job on Sunday purging the TEMP directory, you might want an event trigger if the free space on a C drive drops below ten (10) percent. One or more events can be associated with a job. Defining an event object means you will need to define an event query using WQL (WMI Query Language similar to SQL for database access). Just as SQL requires a database on which to operate WQL requires a *namespace*. A namespace is a local or remote directory that contains various WMI managed data. ActiveBatch uses your security credentials to create the event request.

5 QUICK TOUR (GETTING STARTED)

5.1 Starting the ActiveBatch Admin Application

This Quick Tour focuses on the ActiveBatch Admin application or ABATAdmin for short. ABATAdmin is a Windows GUI utility that is used to access most of the ActiveBatch system (the MMC snap-in is reserved for purely Administrative use).

To run ABATAdmin double-click on the desktop icon (created during the product's installation) or search for the ActiveBatch program group and click on "ActiveBatch Admin". ABATAdmin begins by announcing its startup and then initializes itself for use.

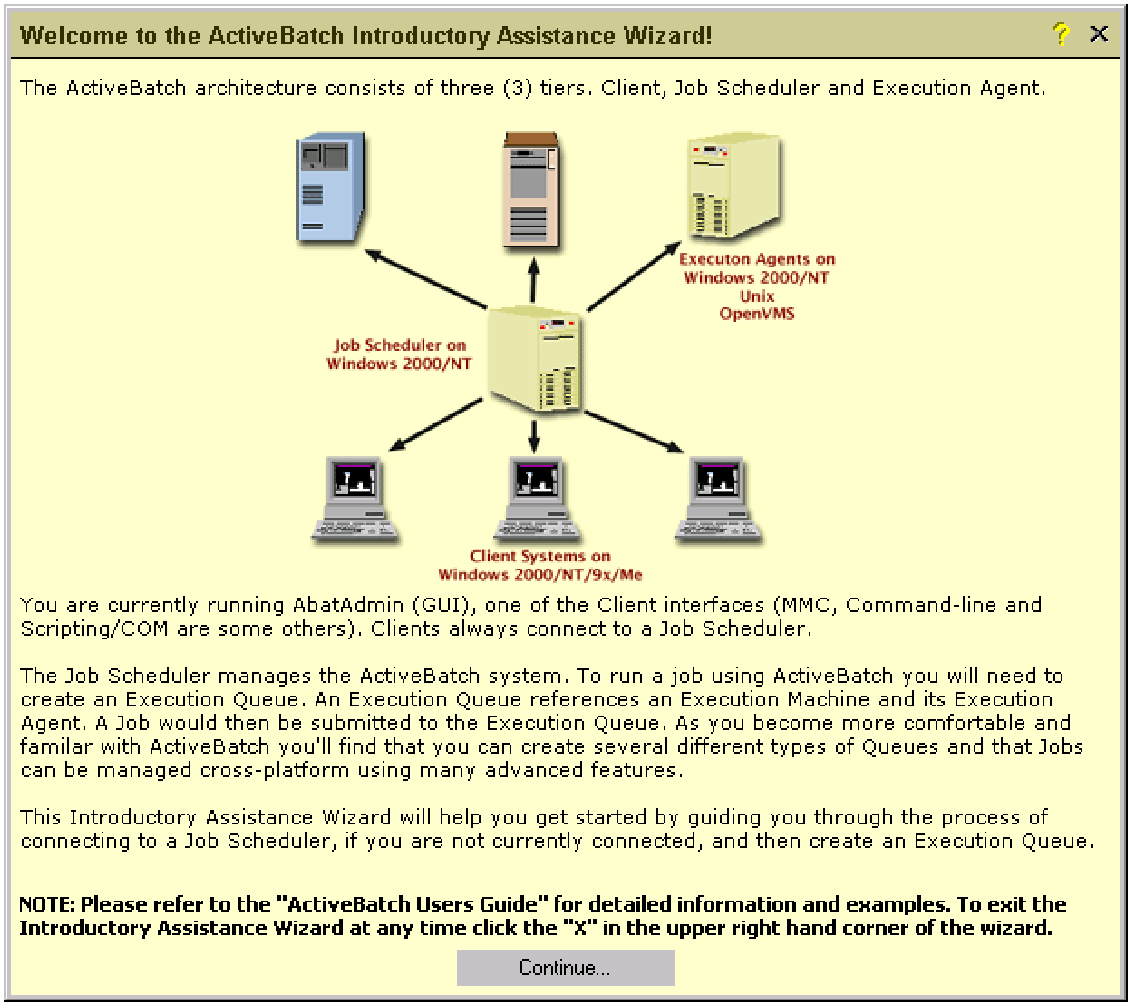


[11-Jul-2000 10:43:29] Cleared log file

Figure 16 - ActiveBatch Admin Utility (Initial)

5.1.1 Introductory Assistance Wizard

For first-time users, ABATAdmin will start the Introductory Assistance Wizard. This wizard will explain some of the basic architecture and terms and then help you connect to a Job Scheduler (usually one that's on the same machine) and create your first Execution Queue so you can try submitting batch jobs.



Welcome to the ActiveBatch Introductory Assistance Wizard!

The ActiveBatch architecture consists of three (3) tiers. Client, Job Scheduler and Execution Agent.

You are currently running AbatAdmin (GUI), one of the Client interfaces (MMC, Command-line and Scripting/COM are some others). Clients always connect to a Job Scheduler.

The Job Scheduler manages the ActiveBatch system. To run a job using ActiveBatch you will need to create an Execution Queue. An Execution Queue references an Execution Machine and its Execution Agent. A Job would then be submitted to the Execution Queue. As you become more comfortable and familar with ActiveBatch you'll find that you can create several different types of Queues and that Jobs can be managed cross-platform using many advanced features.

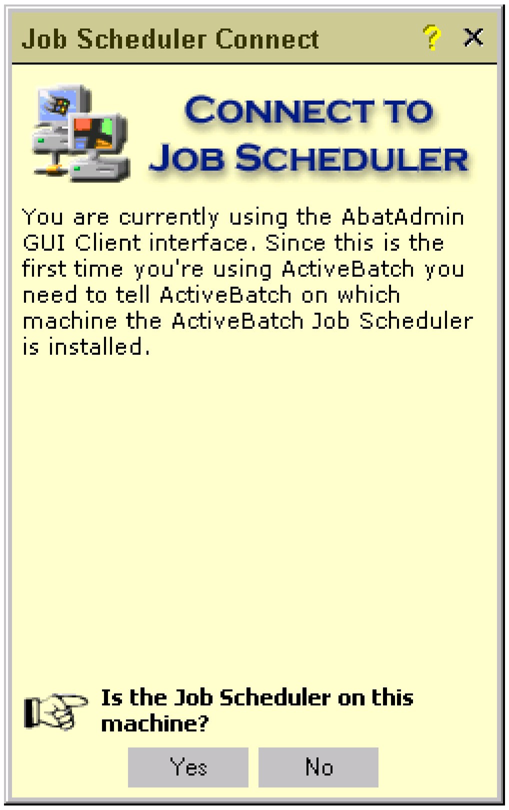
This Introductory Assistance Wizard will help you get started by guiding you through the process of connecting to a Job Scheduler, if you are not currently connected, and then create an Execution Queue.

NOTE: Please refer to the "ActiveBatch Users Guide" for detailed information and examples. To exit the Introductory Assistance Wizard at any time click the "X" in the upper right hand corner of the wizard.

Continue...

Figure 17 - Introductory Assistance Wizard - Startup

You can exit the Wizard at any time by clicking on the “X” in the upper right-hand corner of the wizard. To continue and connect to a Job Scheduler click the *Continue* button at the bottom of the screen.



You are currently using the AbatAdmin GUI Client interface. Since this is the first time you're using ActiveBatch you need to tell ActiveBatch on which machine the ActiveBatch Job Scheduler is installed.

Is the Job Scheduler on this machine?

Figure 18 - Introductory Assistance Wizard - Job Scheduler Connect

The Job Scheduler Connect page is displayed and you are asked whether the Job Scheduler is on this machine or another. Regardless of how you answer you will see the “Connect” dialog box as depicted in Figure 19. If you connect successfully, you can create an Execution Queue using the Queue Wizard as further described in Section 5.3 (which appears later in this Chapter).

You can always request the Introductory Assistance Wizard by clicking on the Help menu for Introductory Assistance Wizard.

5.2 Getting Started Tour

ABATAdmin provides a familiar menu bar as well as an ActiveBatch Tool Bar (for engaging different aspects of the system), an ActiveBatch Navigation Bar and a Navigation View. The initial startup settings for ABATAdmin will display the "ActiveBatch Neighborhood" and "ActiveBatch Navigation Bar and View". Three (3) tabs are provided in this view with each tab presenting the same data organized differently. By default the "Job Schedulers" tab is selected. This tree oriented view of the system will display all connected Job Schedulers and their queues using a drill down orientation. Simply click on the plus sign to expand the tree at that level. If this is the first time you're running ABATAdmin a single entry named "ActiveBatch Neighborhood" will be present. To connect to a Job Scheduler, either right click the ActiveBatch Neighborhood item and select "Connect" or select "JobScheduler/Connect" from the menu bar.

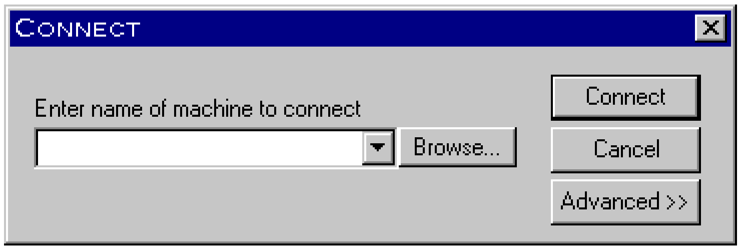


Figure 19 - ABATADmin "Connect" dialog

The Connect dialog allows you to enter a known machine name to connect to or recall a recently entered machine name or browse through the Windows 2000/NT Network Neighborhood. You can enter a machine name, fully qualified machine name or IP address. The machine you select must be running the ActiveBatch Job Scheduler component. Typically the Job Scheduler machine will be in the same domain or security area as your logged on machine. If that's the case, your logon credentials should be enough to authenticate access. However, if you are trying to access the Job Scheduler machine from a machine in which your currently logged on credentials will not grant you access, you will need to click on the "Advanced" button. Enter a username/password that can be properly authenticated for the selected machine. The username syntax supports the additional specification of a domain or machine (e.g. *domain\username* or *machine\username*).

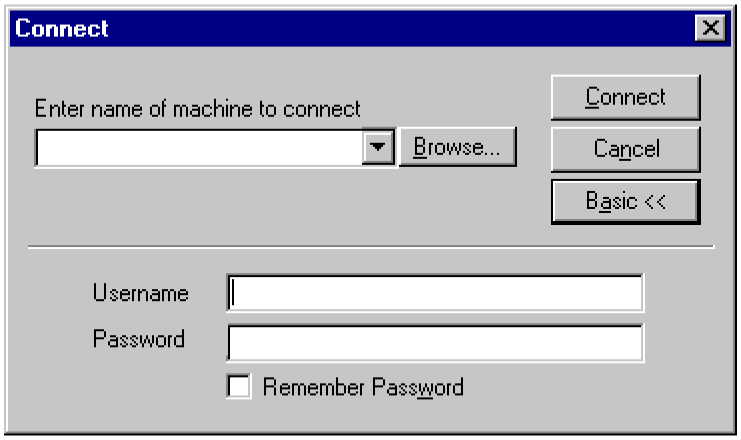
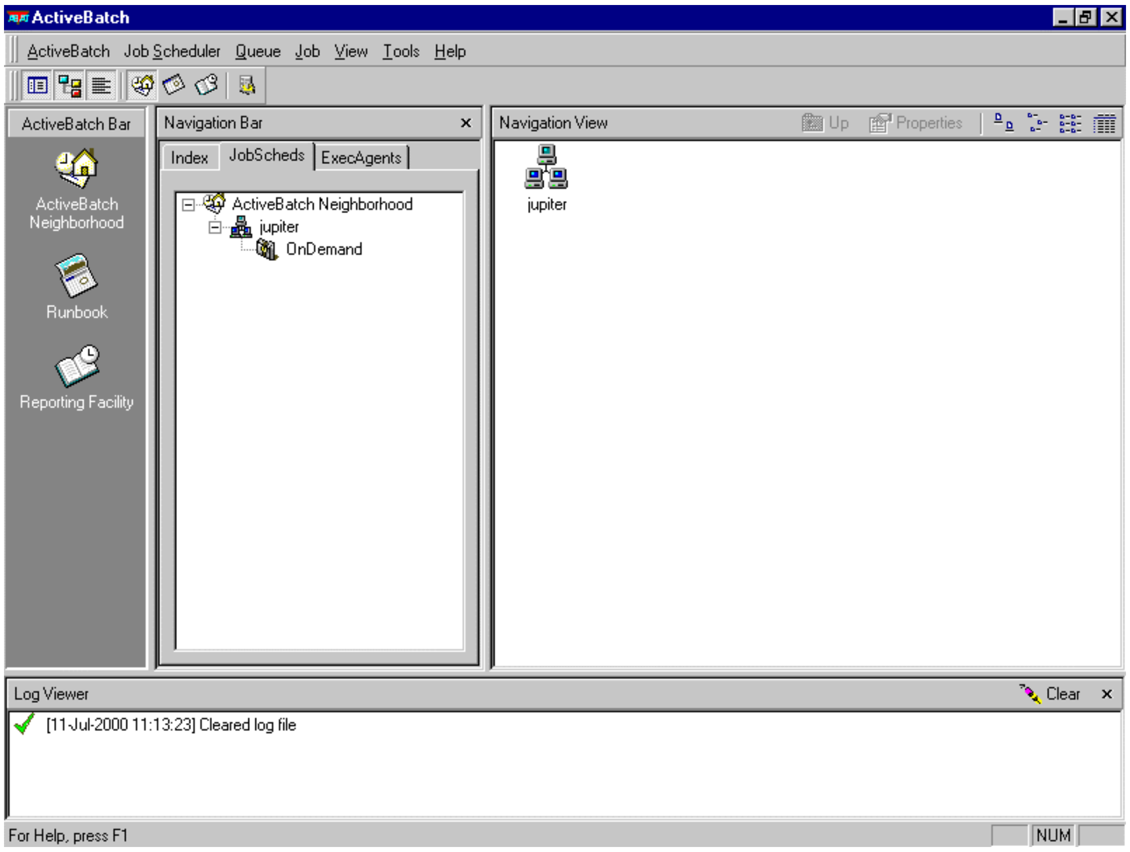


Figure 20 - Connect (Advanced) Dialog

Clicking the OK button will cause ABATAdmin to attempt to connect to the selected machine. If the connection succeeds, queue information will be displayed under the Job Scheduler machine name."



```
[11-Jul-2000 11:13:23] Cleared log file
```

Figure 21 - Quick Tour Sample Connection

This is a good time to point out the "Log Viewer" window at the bottom. All success and error messages are displayed here.

While an in-depth explanation of ABATAdmin is provided in Section 6, you can specify various in-context operations by either right clicking on a tree item or selecting the operation from the Menu bar.

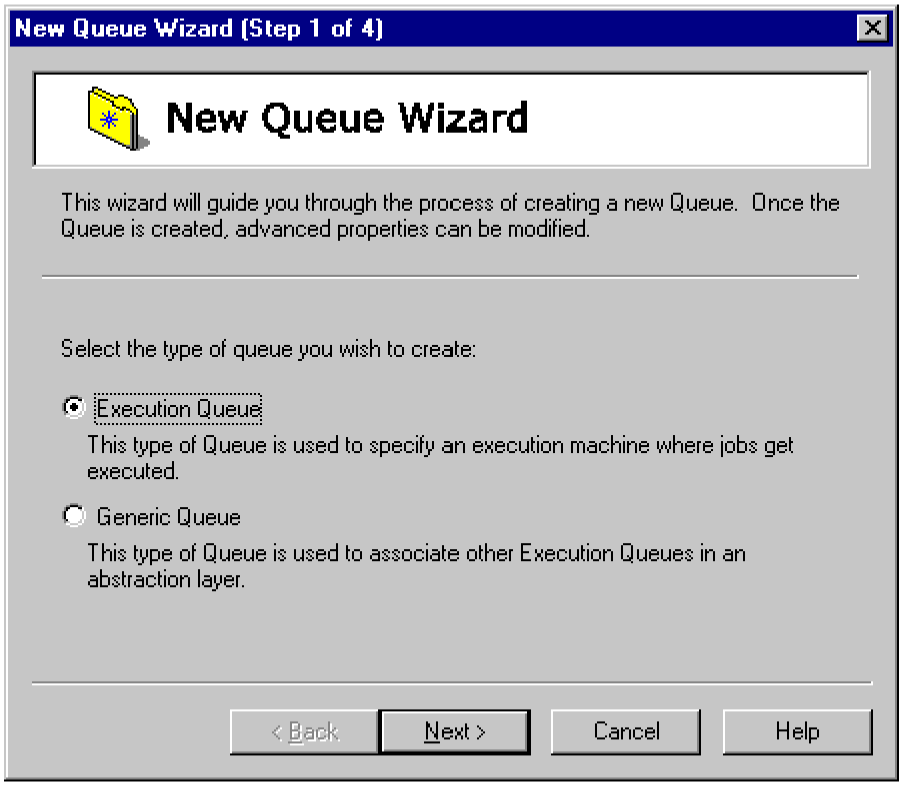
5.3 Creating an Execution Queue

With ActiveBatch started, you're ready to start creating queues. By default, only members of the *Administrators* group can create queues, so if other users on your machines are to be able to create queues, you will need to use the MMC snap-in to provide that permission.

On a brand new ActiveBatch system, one queue named the *OnDemand* queue will be displayed. We are not going to reference that queue until Section 6.6, so let's ignore it for now.

Since we want to quickly get things running, we will create an Execution Queue. An Execution Queue represents a machine that we will want to run jobs on.

To create an execution queue, right click on the Job Scheduler item and select "New Queue (Wizard)" or select "JobScheduler/New Queue (Wizard)" from the Menu bar.



This wizard will guide you through the process of creating a new Queue. Once the Queue is created, advanced properties can be modified.

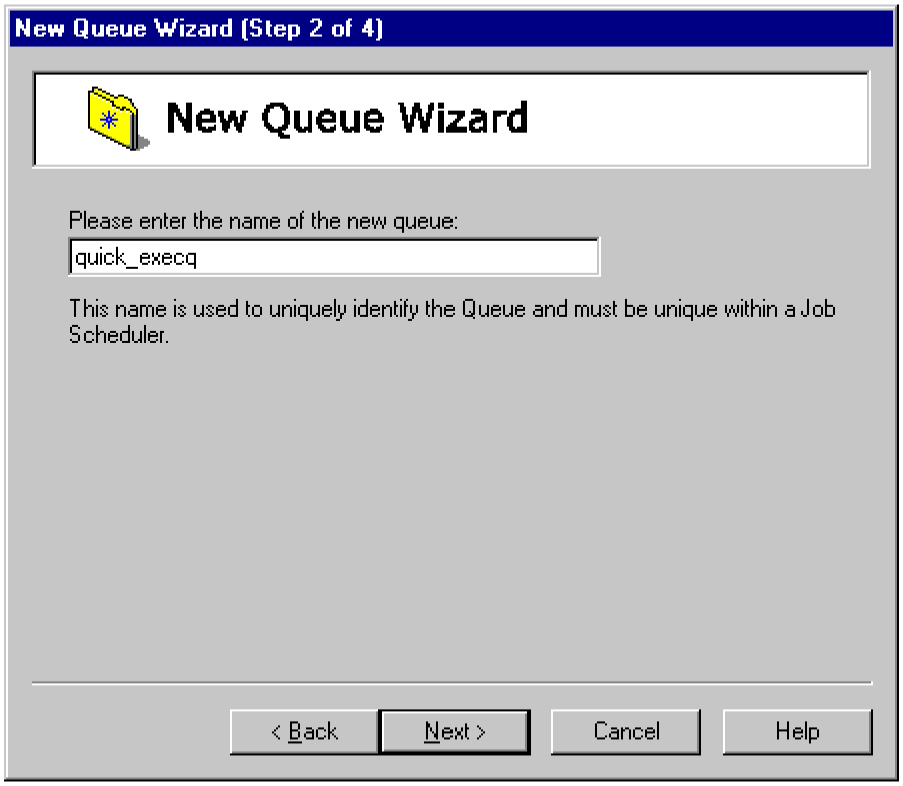
Select the type of queue you wish to create:

[✓ Execution Queue]
This type of Queue is used to specify an execution machine where jobs get executed.

Generic Queue
This type of Queue is used to associate other Execution Queues in an abstraction layer.

Figure 22 - New Queue Wizard – Page 1

Figure 22 shows the first page of the New Queue Wizard. This wizard will quickly let you create an Execution Queue. You must indicate whether you want to create a Generic or Execution Queue. Click the Execution Queue radio button to select that type of queue and then click the "Next" button.


New Queue Wizard (Step 2 of 4)

New Queue Wizard

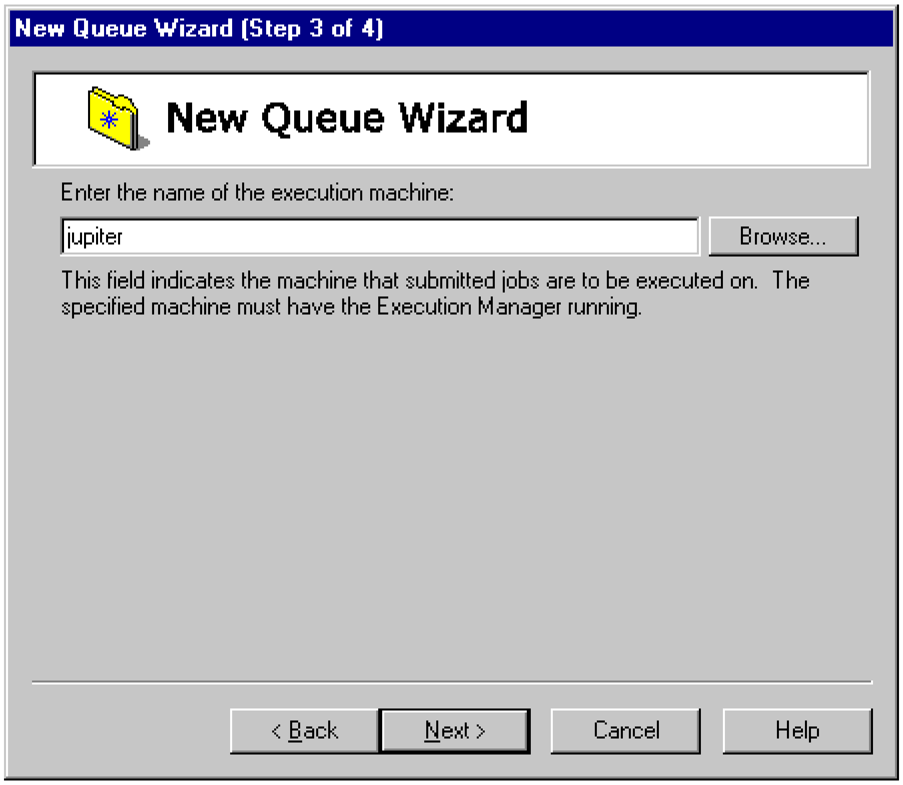
Please enter the name of the new queue:

quick_execq

This name is used to uniquely identify the Queue and must be unique within a Job Scheduler.

Figure 23 - New Queue Wizard – Page 2

This page requires that you enter the name of the Execution Queue you want to create. This name must be unique within a Job Scheduler. While you can name your Execution Queue anything you like you might want to consider naming the queue with at least a portion of the actual machine name the queue is to point to. This will make things easier to follow. For the purposes of our Quick Tour, let's name the queue "quick_execq" and then press the "Next" button.


New Queue Wizard (Step 3 of 4)

New Queue Wizard

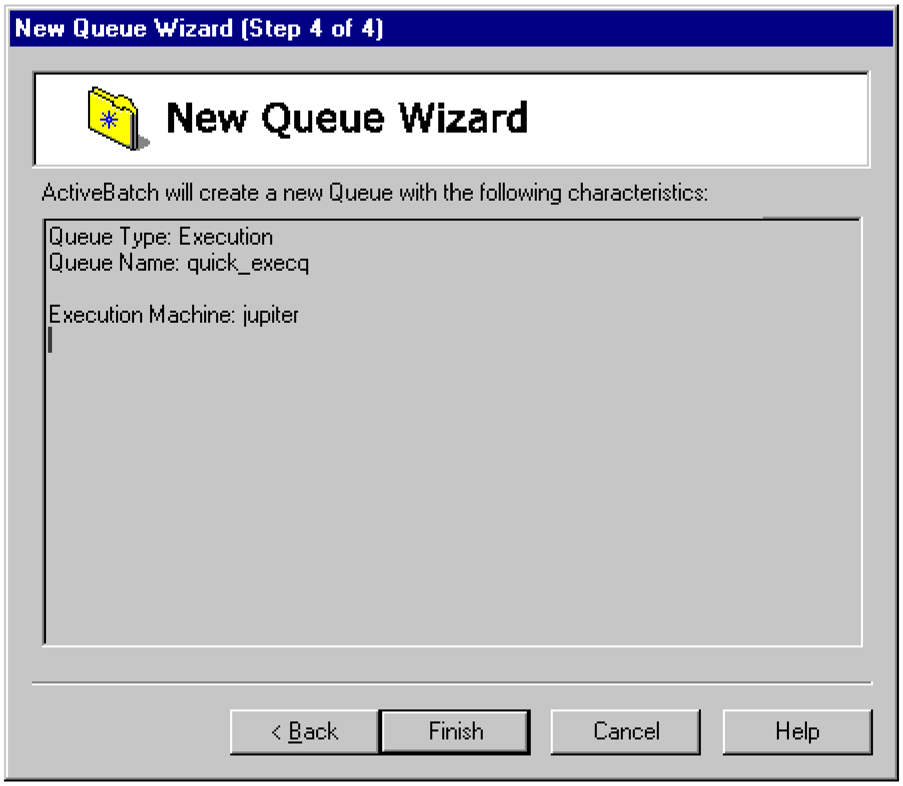
Enter the name of the execution machine:

jupiter

This field indicates the machine that submitted jobs are to be executed on. The specified machine must have the Execution Manager running.

Figure 24 - New Queue Wizard – Page 3

This page requires that you enter the name of the machine that is to be represented by this Execution Queue. In other words, when a job is submitted to this queue, the job is to run on this machine. Remember that an Execution Agent must be running on this machine for jobs in this queue to successfully execute. You can enter the machine using any valid and accepted transport syntax. Typically a machine name or IP address will be entered. You can also browse the NT Network Neighborhood and select a machine by clicking on the *Browse* button. Click the "Next" button to continue to the final page. In our example we'll be using a machine named "Jupiter".



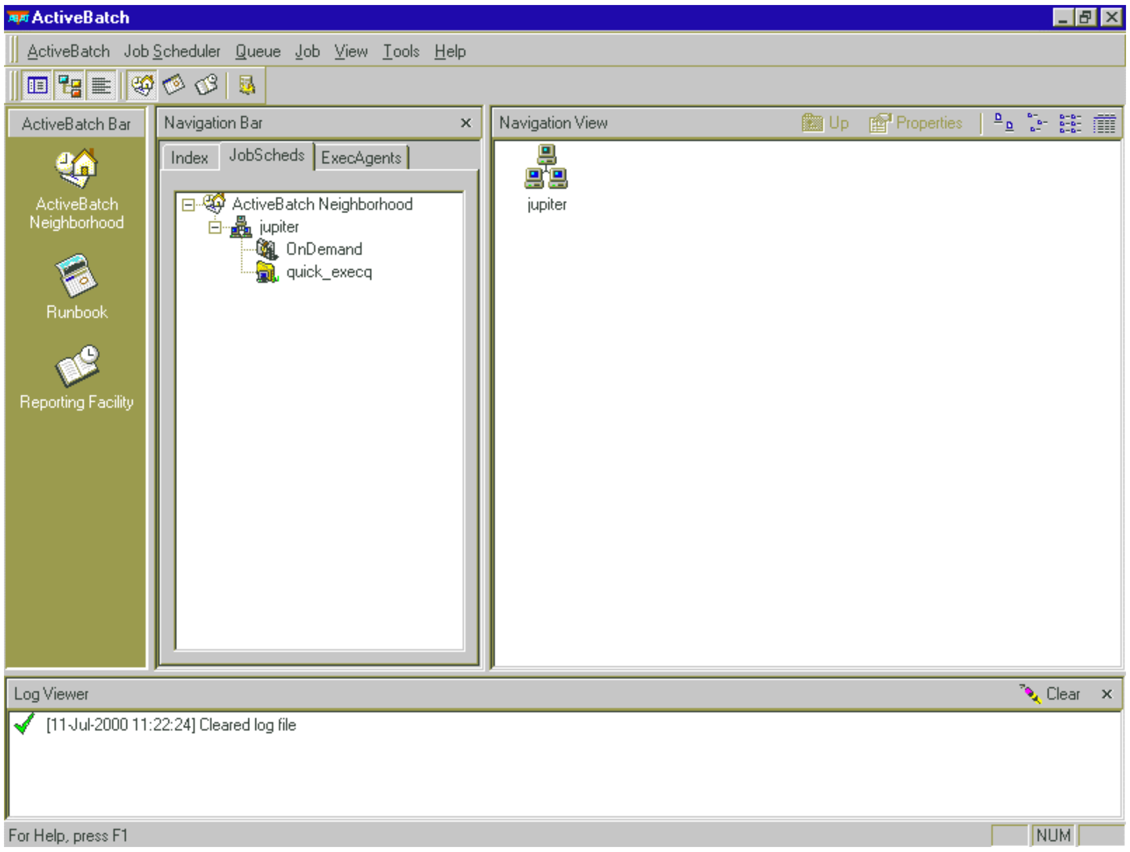
ActiveBatch will create a new Queue with the following characteristics:

```
Queue Type: Execution
Queue Name: quick_execq

Execution Machine: jupiter
```

Figure 25 - New Queue Wizard – Last Page

This confirmation page displays all the information you entered and asks that you confirm the creation of the Execution Queue using the values displayed. Clicking on "Finish" will cause the operation to occur. If you need to go back and change any values simply press the "Back" button and browse through the previously entered pages. Figure 26 displays what the Navigation Bar/JobScheds view would look like after the queue has been created.

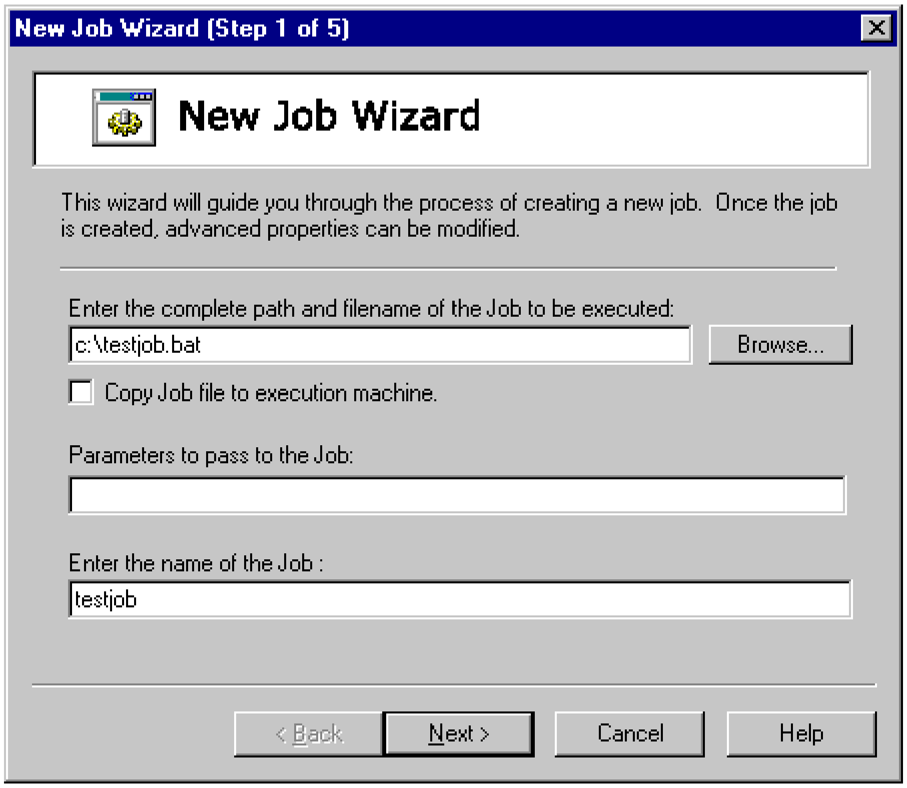


```
[11-Jul-2000 11:22:24] Cleared log file
```

Figure 26 - Quick Tour New Queue Wizard

5.4 Submitting a Simple Job

Now that we've created an Execution Queue, let's submit a job to it. Remember that a job is typically a script file or program that we want to execute in an unattended mode. The "simple" job we refer to won't initially use many of the ActiveBatch facilities. This will be a job that we want to run now. To submit a job, right click on the Execution Queue we created (quick_execq) and select "New Job (Wizard)" or "Queue/New Job (Wizard)" from the menu bar.


New Job Wizard (Step 1 of 5)

New Job Wizard

This wizard will guide you through the process of creating a new job. Once the job is created, advanced properties can be modified.

Enter the complete path and filename of the Job to be executed: c:\testjob.bat

☐ Copy Job file to execution machine.

Parameters to pass to the Job:

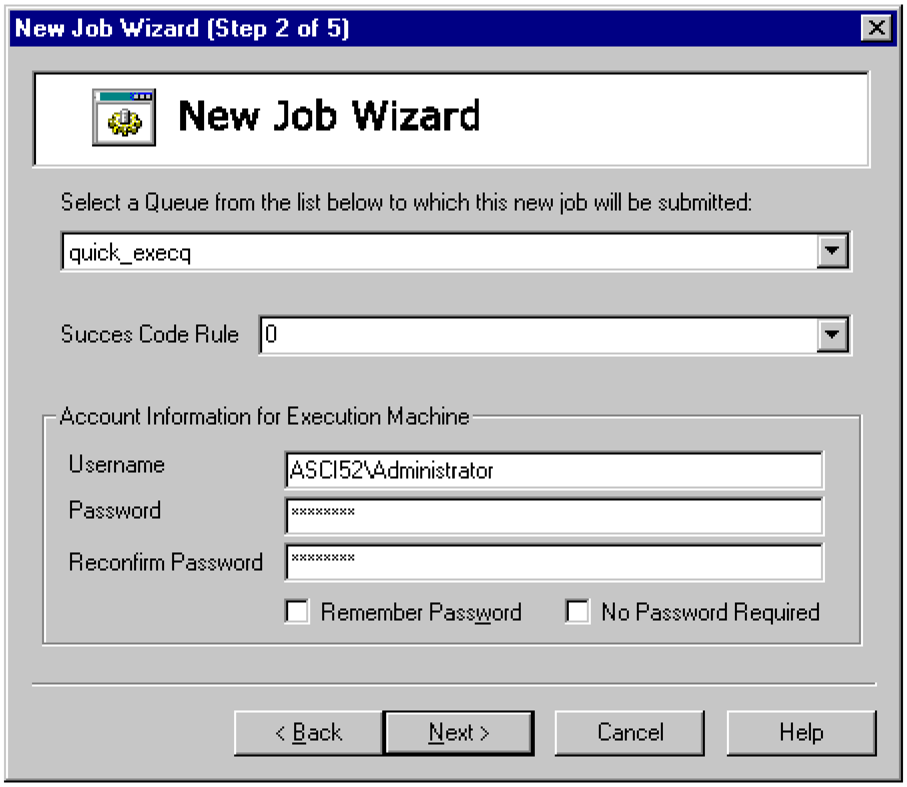
Enter the name of the Job : testjob

Figure 27 - New Job Wizard – Page 1

The first page of the New Job Wizard (as seen above in Figure 27) requires you to enter the file specification of the job's script file or program you wish to run. It is critically important to remember that the file must be accessible through the file specification you enter from the execution machine's point of view. So don't enter C:\testfiles unless you intend for the job to be locally executed on the client machine. If your job file is local to your machine and must be copied over to the Execution machine then check the box *Copy Job to Execution Machine*. Typically, for remote machines, you will specify either a UNC specification allowing complete unambiguous direction to where the file is located or the file will be located locally on the executing machine. It is also important to understand that the file will be accessed with your security context. So you must be able to read or have execute permission access to the file. You can enter the file specification or use the Browse button to locate and then select the file you need.

The second field on the page allows you to enter input parameters that will be specified with the job when the job step is executed. These parameters are the same as those entered with a command-line based program to further refine or request actions. Your ability to specify input parameters and parameter usage is completely up to the job author. Some jobs may support input parameters some may not.

The third field is “Job Name”. To prevent confusion, all jobs in ActiveBatch should have a unique name. By default, ActiveBatch will use the filename portion of the job file specification as a Job Name. As you enter your job file specification, the job name will appear at the same time. You can always edit the field to change the job name to whatever you wish. Please note that when entering non-Windows file specifications (e.g. UNIX, OpenVMS) the job name may contain portions of the file specification that you would prefer not be a part of the job name. If so, you can always edit the job name to whatever you desire. Click “Next” to continue to the next page.



Select a Queue from the list below to which this new job will be submitted:
quick_execq

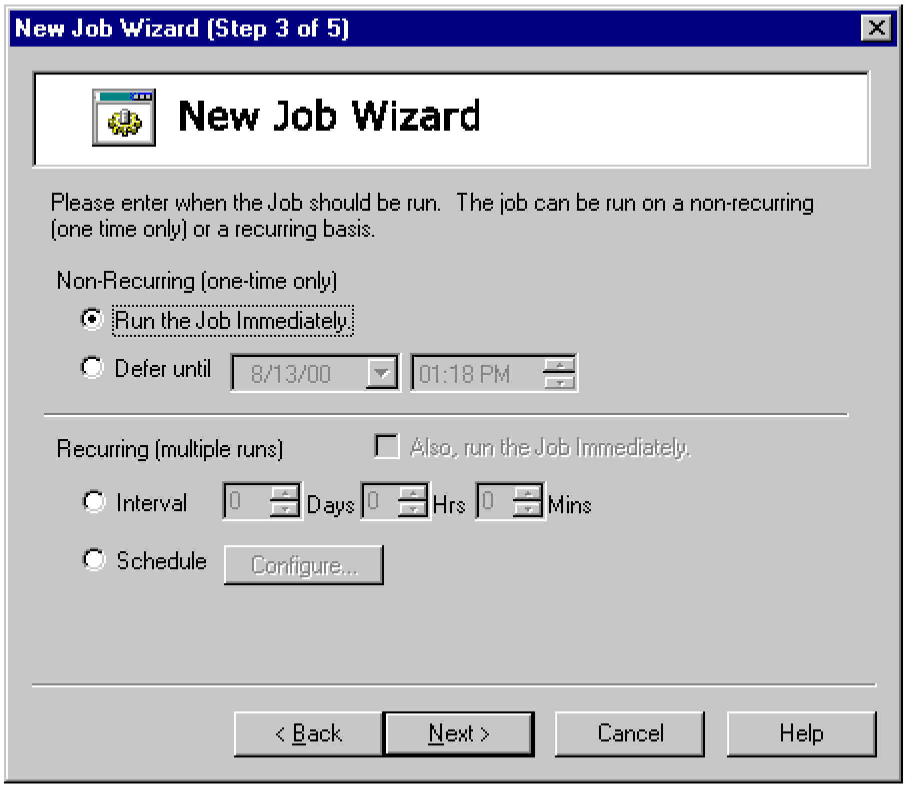
Succes Code Rule: 0

Account Information for Execution Machine
Username: ASCI52\Administrator
Password: ********
Reconfirm Password: ********
☐ Remember Password ☐ No Password Required

Figure 28 - New Job Wizard – Page 2

The second page of the New Job Wizard prompts for a queue to which the job is to be submitted. This can be any queue as listed in the pull down list. For the purposes of this Quick Tour, select "quick_execq". When a job is run, there must be a way of determining whether the job ran successfully or not. The wizard provides a "Success Code Rule" to help make this determination. The default exit code for success is zero (0). If your job uses other codes for success please indicate them here. XLNT users will want to enter NTMSG to enable NT message code handling (the upper 2 bits determine severity). You can use the Success Code Rule dropdown to make your selection (see Section 6.5.3.7).

ActiveBatch runs your jobs under your security context. By default, your logon username will be filled in where the field "Username" is specified (a *domain\username* can also be specified). You must then enter your password in the "Password" field. This password is not echoed for security reasons. You can tell ActiveBatch to save the password associated with this username by clicking the "Remember Password" checkbox. Remember that the username and password entered must be valid on the execution machine this job is expected to run on. For this Quick Tour, the execution machine is the same machine as this one so simply enter your password associated with this username account. Click “Next” to continue to the next page.



Please enter when the Job should be run. The job can be run on a non-recurring (one time only) or a recurring basis.

Non-Recurring (one-time only)

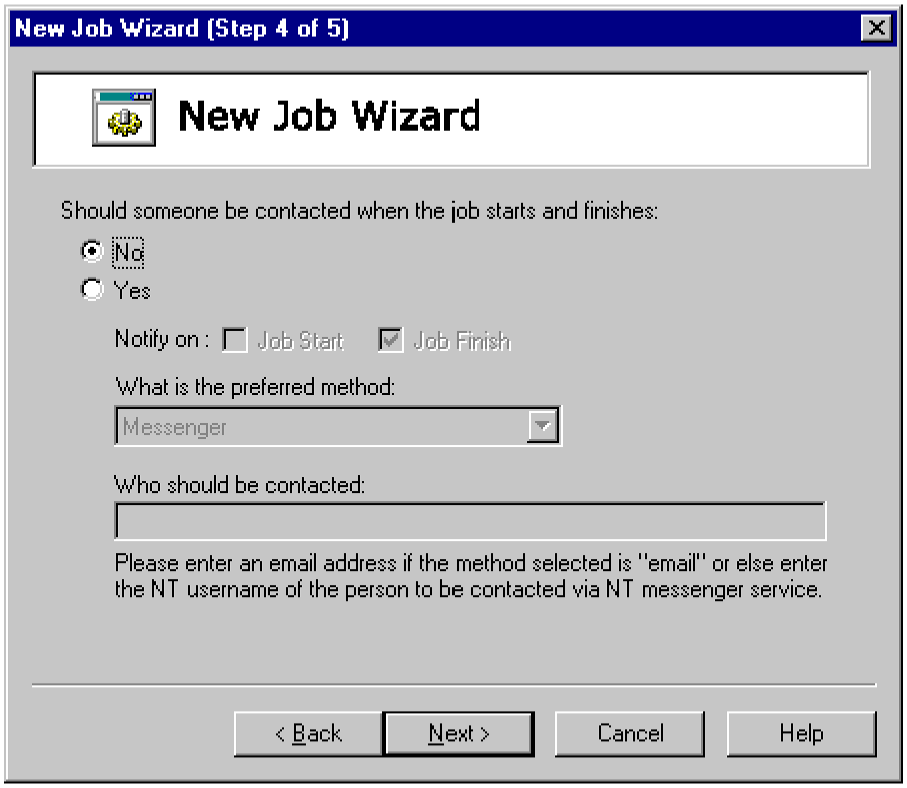
[✓ Run the Job Immediately]
Defer until 8/13/00 01:18 PM

Recurring (multiple runs)   Also, run the Job Immediately.

Interval 0 Days 0 Hrs 0 Mins
Schedule Configure...

Figure 29 - New Job Wizard – Page 3

The third page of the New Job Wizard concerns the scheduling of the job. You can select recurring or non-recurring type scheduling. Recurring indicates that the job is expected to execute more than once on some type of schedule. A non-recurring job is one that is expected to execute only once. Non-Recurring choices allow for immediate or deferred execution. Recurring choices include either immediate, time interval or a schedule for job execution. For the purposes of our Quick Tour, select non-recurring immediate execution by clicking on that radio button. Then click “Next” to continue to the next page.


New Job Wizard (Step 4 of 5)

New Job Wizard

Should someone be contacted when the job starts and finishes:

[✓ No] Yes

Notify on : Job Start [✓ Job Finish]

What is the preferred method:

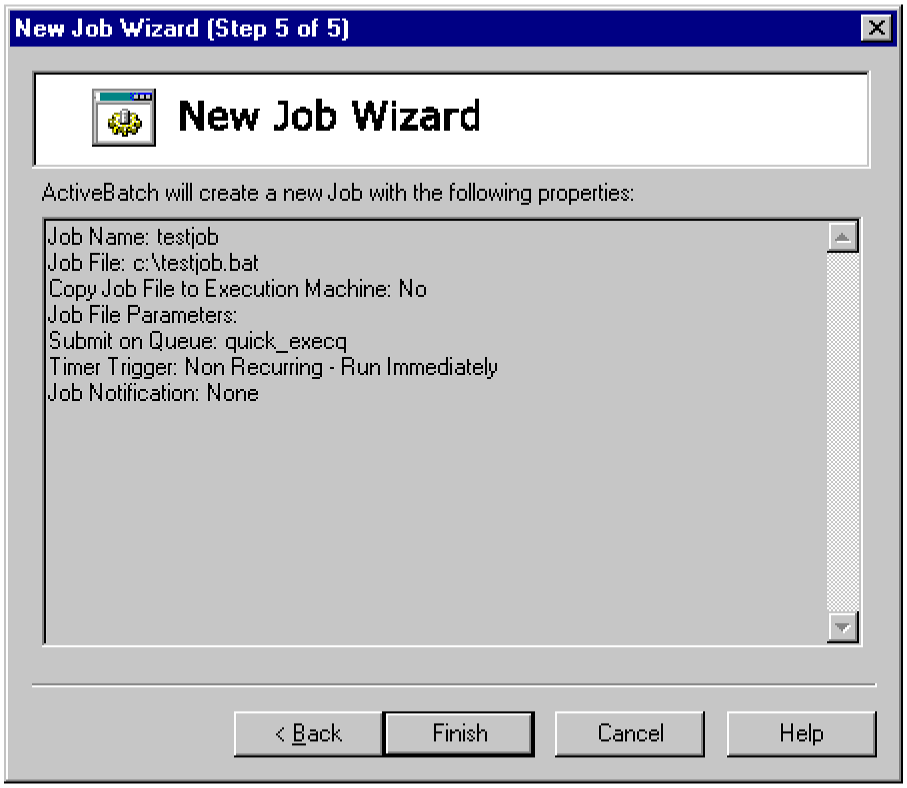
Messenger

Who should be contacted:

Please enter an email address if the method selected is "email" or else enter the NT username of the person to be contacted via NT messenger service.

Figure 30 - New Job Wizard – Page 4

The fourth page of the New Job Wizard concerns the Job's Contact Information. ActiveBatch allows you to specify information for a person to be contacted when the job starts and/or finishes. The "No" selection indicates that no contact information is desired. For the purposes of this Quick Tour, let's assume "No" and click "Next" to continue to the next page. Don't worry we will cover the "Yes" side of this during the Quick Tour.



ActiveBatch will create a new Job with the following properties:

```
Job Name: testjob
Job File: c:\testjob.bat
Copy Job File to Execution Machine: No
Job File Parameters:
Submit on Queue: quick_execq
Timer Trigger: Non Recurring - Run Immediately
Job Notification: None
```

Figure 31 - New Job Wizard – Last Page

The last page of the New Job Wizard summarizes all of the previous answers given to allow you to confirm and submit the job, or go back and change some of your answers. You can also "Cancel" the job submission altogether. Clicking on the "Finish" button causes the job to be submitted to the queue you specified.

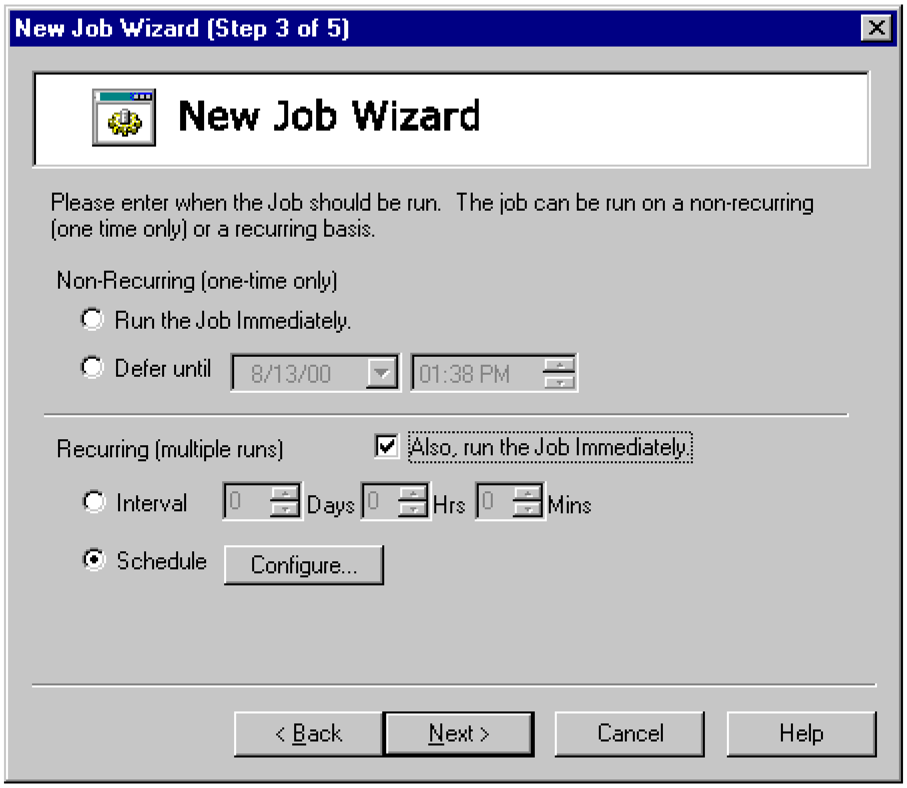
When the job is submitted you should see the job appear as an icon in the "Navigation View" area of the window. This view can display icons or textual information. The icon contains color indicating the overall state of the job. Once the job has been executed the job icon will disappear and the job will become "history". That's both figurative and literal. A non-recurring job is removed from the active system when the job completes execution. Don't worry though you can always use the "Runbook View", on the ActiveBatch Bar, to examine previously executed jobs. We'll do that after we submit a slightly more complicated job as described in the next section.

5.5 Submitting a Scheduled Job

The previous example submitted a simple immediate job for execution. This example will take you through the process of submitting a job on a recurring schedule.

For this example, we will continue to use the "New Job (Wizard)" operation. Select the "quick_execq" Execution Queue and right click to select the "New Job (Wizard)".

We've already described the first few pages which are pretty self explanatory so let's cut to the third page that describes the frequency and time the job is expected to execute.


New Job Wizard (Step 3 of 5)

New Job Wizard

Please enter when the Job should be run. The job can be run on a non-recurring (one time only) or a recurring basis.

Non-Recurring (one-time only)

Run the Job Immediately.

Defer until 8/13/00 01:38 PM

Recurring (multiple runs) [✓ Also, run the Job Immediately.]

Interval 0 Days 0 Hrs 0 Mins

[✓ Schedule] Configure...

Figure 32 - New Job Wizard – Recurring/Immediate

This time lets select the "Schedule" under the "Recurring (multiple runs)" section of the page. Selecting the "Schedule" radio button causes the "also run immediately" and "Configure" buttons to become active. Let’s take the opportunity to check the “Also, run the Job Immediately” checkbox. Pressing the “Configure” button will cause the scheduling dialog to appear.

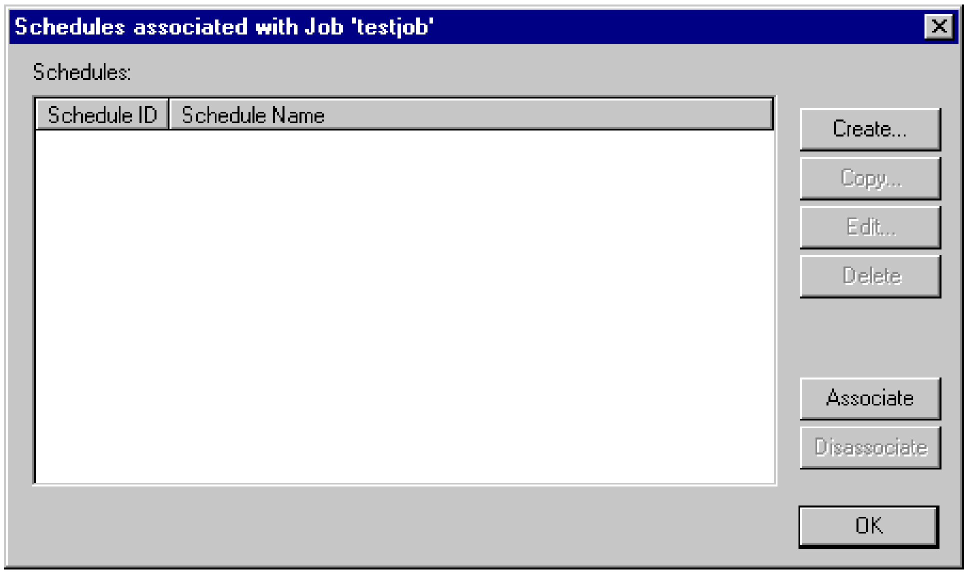
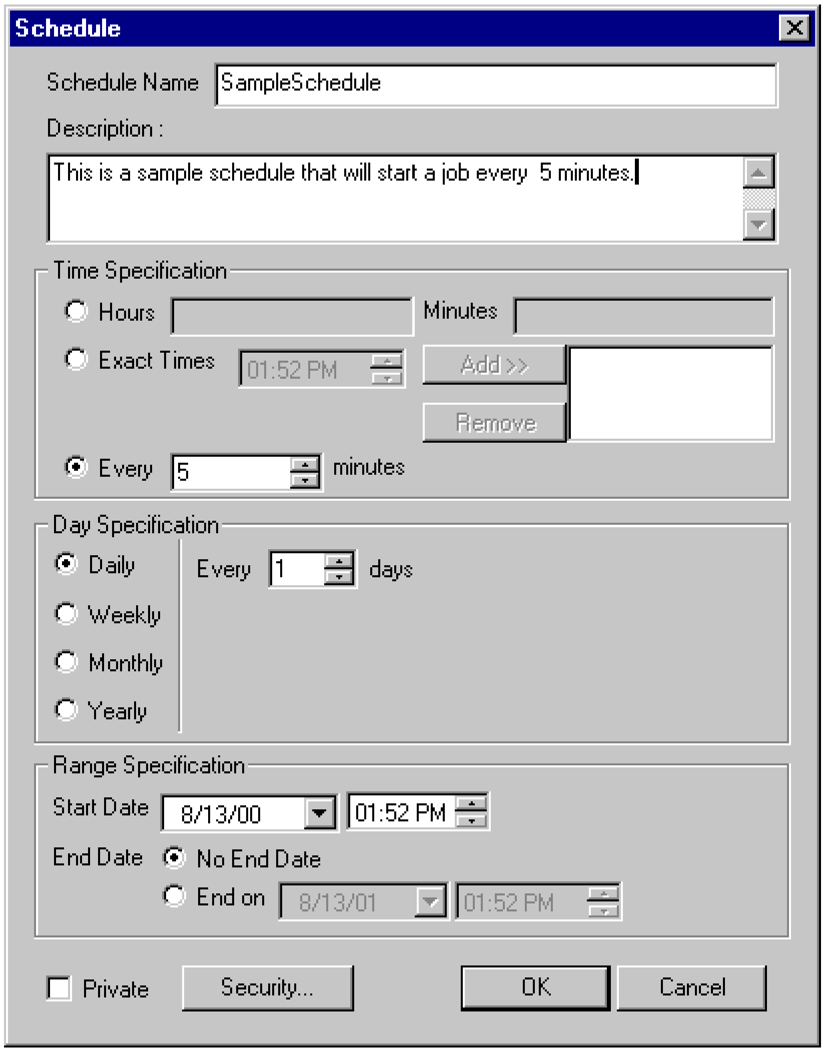


Figure 33 - New Job Wizard – Schedule Dialog

ActiveBatch Schedules are objects that can be shared. If you need several jobs to run on a Friday at 7pm you won’t have to create a special schedule for each job. Rather you create one schedule and simply attach the schedule to the job. The “Schedule” dialog (Figure 33) lists all schedules that have been associated with this job. Since this no schedules currently exist, Figure 34 is displayed over the dialog box.


Schedule

Schedule Name: SampleSchedule

Description: This is a sample schedule that will start a job every 5 minutes.

Time Specification

( ) Hours   Minutes

( ) Exact Times 01:52 PM Add >> Remove

(●) Every 5 minutes

Day Specification

(●) Daily Every 1 days

( ) Weekly

( ) Monthly

( ) Yearly

Range Specification

Start Date 8/13/00 01:52 PM

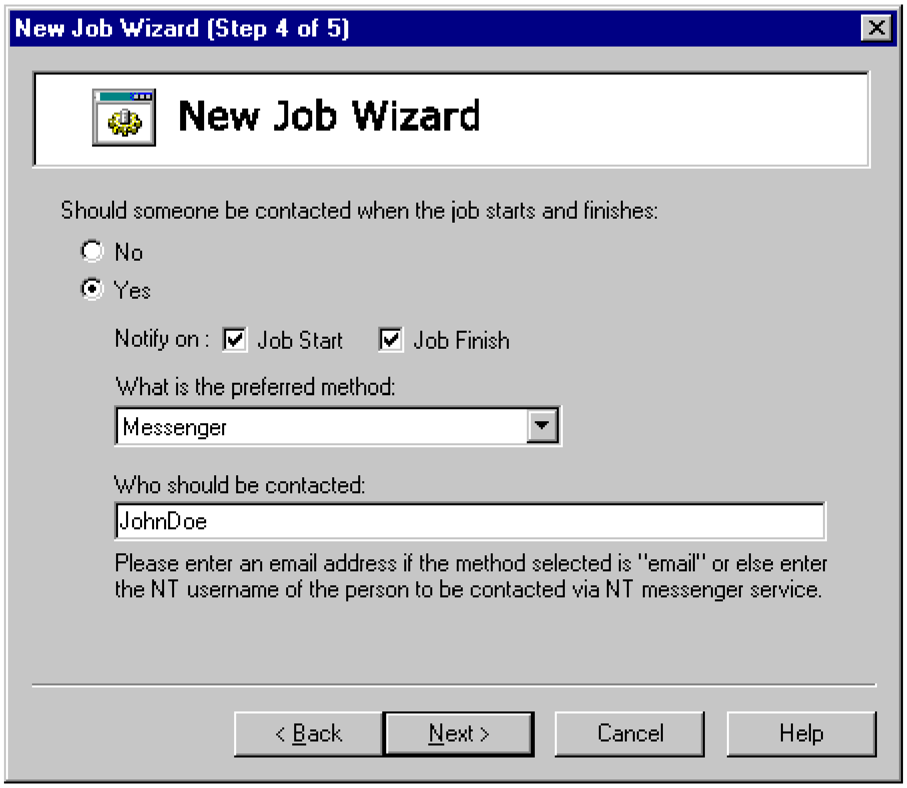
End Date (●) No End Date

( ) End on 8/13/01 01:52 PM

☐ Private

Figure 34 - New Job Wizard – Schedule Create

Figure 34 depicts a completed schedule. First, we need to name our schedule. All ActiveBatch Schedules have unique names. Once you've named your schedule (let's use "SampleSchedule" for this one). You can enter a description of the schedule to further explain what it is supposed to do. You'll notice that the remaining portion of the dialog box is divided into three areas. The first area is for "Time" specifications. This area governs what time the job will run on selected dates or days. The second area is for "Day" or date specification. This area governs what day(s) or date(s) you want the job to run. The third area governs the effectiveness of the Schedule itself. In other words, when is the schedule active and when does it expire. Since this is a “Quick Tour” we’re going to keep things simple. Section 6.5.3.2 goes into much greater detail on what types of schedules you can create. One important point to add though is that a job can have multiple schedules. So don’t think you have to cram every possible run contingency into a single schedule. You don’t have to. For this example, let’s click on the “Every” radio button within the “Time” area and specify 5 minutes. This means that we want this job to execute every five (5) minutes (and on a :05 time boundary). Since “Daily” is selected by default, this job, if we let it, would run every 5 minutes on every day. Click “OK” to confirm, create and associate the schedule. Your named schedule will now appear on the “Schedule” list. Click “OK” to continue back to the “Scheduling” page. Click “Next” to continue to the next page.



Should someone be contacted when the job starts and finishes:

No [✓ Yes]

Notify on : ✓ Job Start ✓ Job Finish

What is the preferred method: Messenger

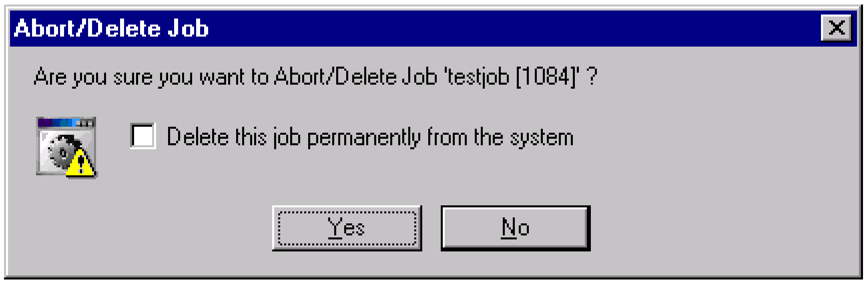
Who should be contacted: JohnDoe

Please enter an email address if the method selected is "email" or else enter the NT username of the person to be contacted via NT messenger service.

Figure 35 - Job Wizard Notification Page

This time let’s specify the Job’s contact information. Click the “Yes” radio button and select the “Messenger” notification method. Your machine name will appear by default. Then select when you wish to be notified (Job Start and Finish). Complete the remaining pages to submit your job.

You can watch the "Navigation View" to see your job start and finish execution. You should let it execute at least once so we have some job history to look at. Each time the job executes you should receive an NT Messenger message box indicating the job is starting execution and then another one that the job has completed (and what the overall status was). To abort and/or delete the job, simply click on the job icon (in the Navigation View) and then right click to "Abort/Delete" the job. A confirmation box will appear to ensure you really want the job aborted or deleted.


Abort/Delete Job

Are you sure you want to Abort/Delete Job 'testjob [1084]' ?

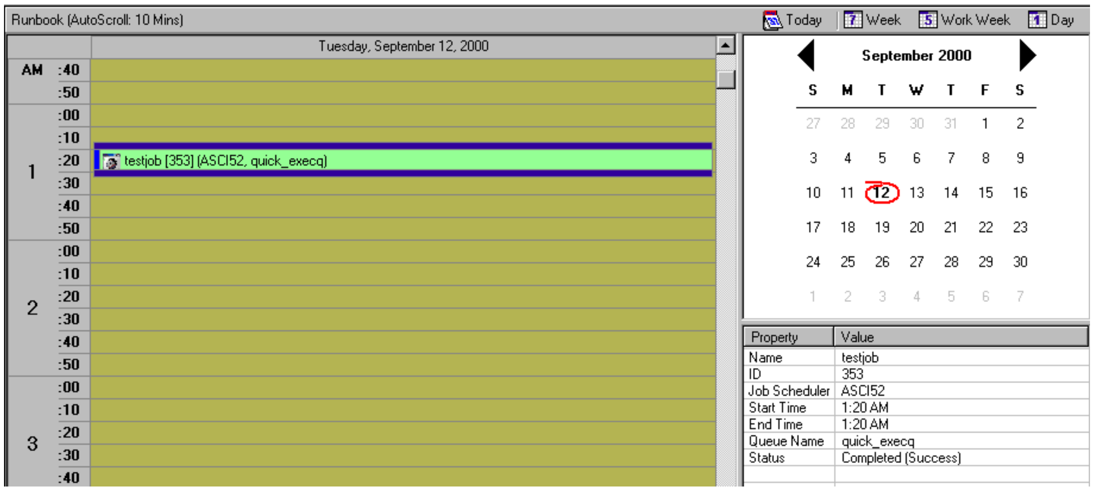
☐ Delete this job permanently from the system

Figure 36 - Quick Tour Delete Job Message Box

5.6 Using the “Views”

If you’ve followed this “Quick Tour” section you should have submitted two jobs. When the jobs are pending or executing they are part of the active portion of the system. When a job has completed, it becomes a part of the history portion of the system. To view past jobs that have executed, you need to enter the “Runbook” view. This view provides a seamless and transparent approach to viewing jobs regardless of whether they have already run, are pending or executing within the system, or whether they are scheduled to run in the future.

The “Runbook” view is day or date oriented. You can select a specific day, workweek or week to view. The left hand portion of the “Runbook” shows the specific period you are interested in viewing. The right portion shows the month and dates you can display. If you examine the “Runbook” tool bar, you’ll notice that you can quickly select “7 Week”, “5 Work Week” or “1 Day”. The “Today” button switches you to today’s date regardless of what date you are actually viewing. For the purposes of the Quick Tour, let’s select “Today” and then “1 Day” view. The left hand portion appears as a page from a diary. Time is listed down the left side of the frame and space for job entries take up the remaining portion of the frame. You can use the scroll bar along the right side of the frame to scroll through the day.



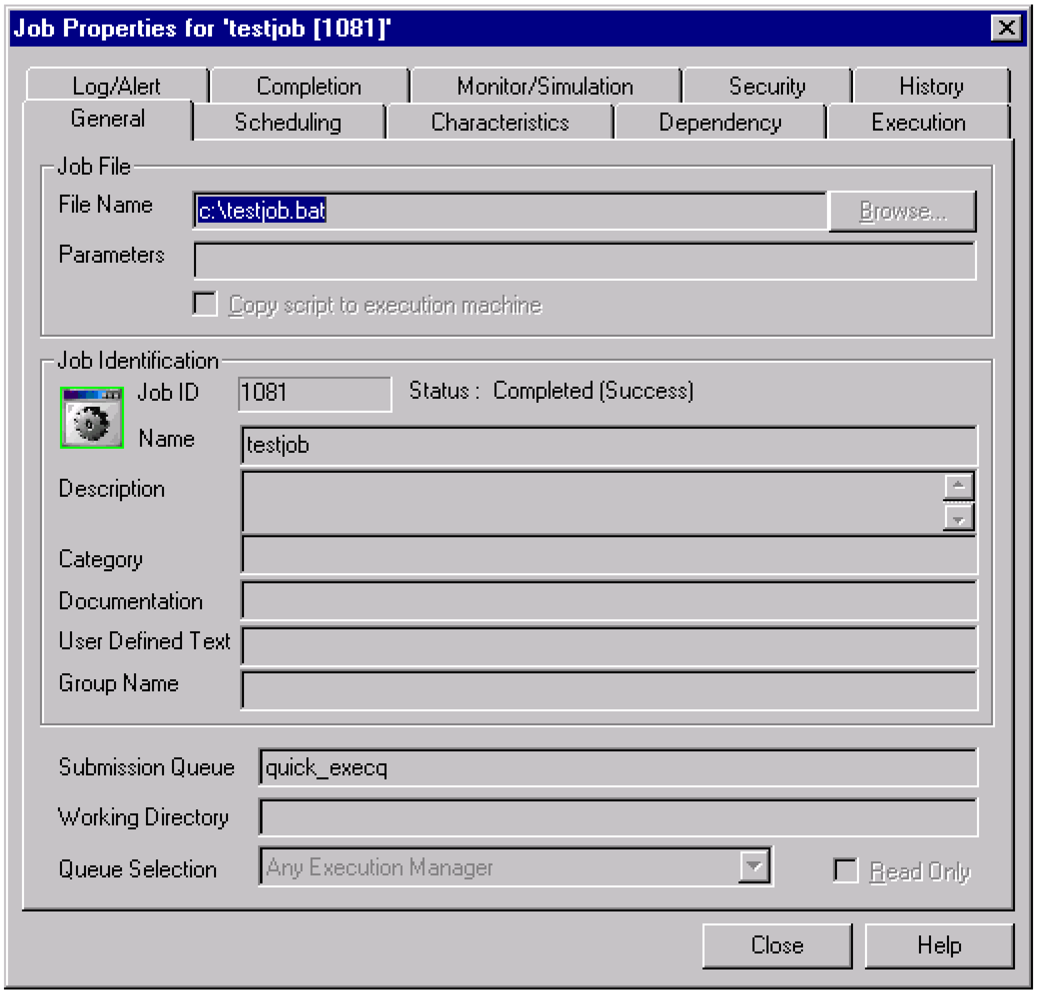
testjob [353] (ASCI52, quick_execq)

S	M	T	W	T	F	S
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
1	2	3	4	5	6	7

Property	Value
Name	testjob
ID	353
Job Scheduler	ASCI52
Start Time	1:20 AM
End Time	1:20 AM
Queue Name	quick_execq
Status	Completed (Success)

Figure 37 - Quick Tour Runbook View (Today/1Day)

For the purposes of this Quick Tour, locate the 2 jobs you ran and click on the first one. You should notice several job properties appear in a small window under the month time frame. This window displays several frequently accessed job properties. If you want to see all the properties and history of the job, just double-click on the job entry in the Job Properties frame.



Job File

File Name: c:\testjob.bat

Parameters:

☐ Copy script to execution machine

Job Identification

Job ID: 1081 Status : Completed (Success)

Name: testjob

Description:

Category:

Documentation:

User Defined Text:

Group Name:

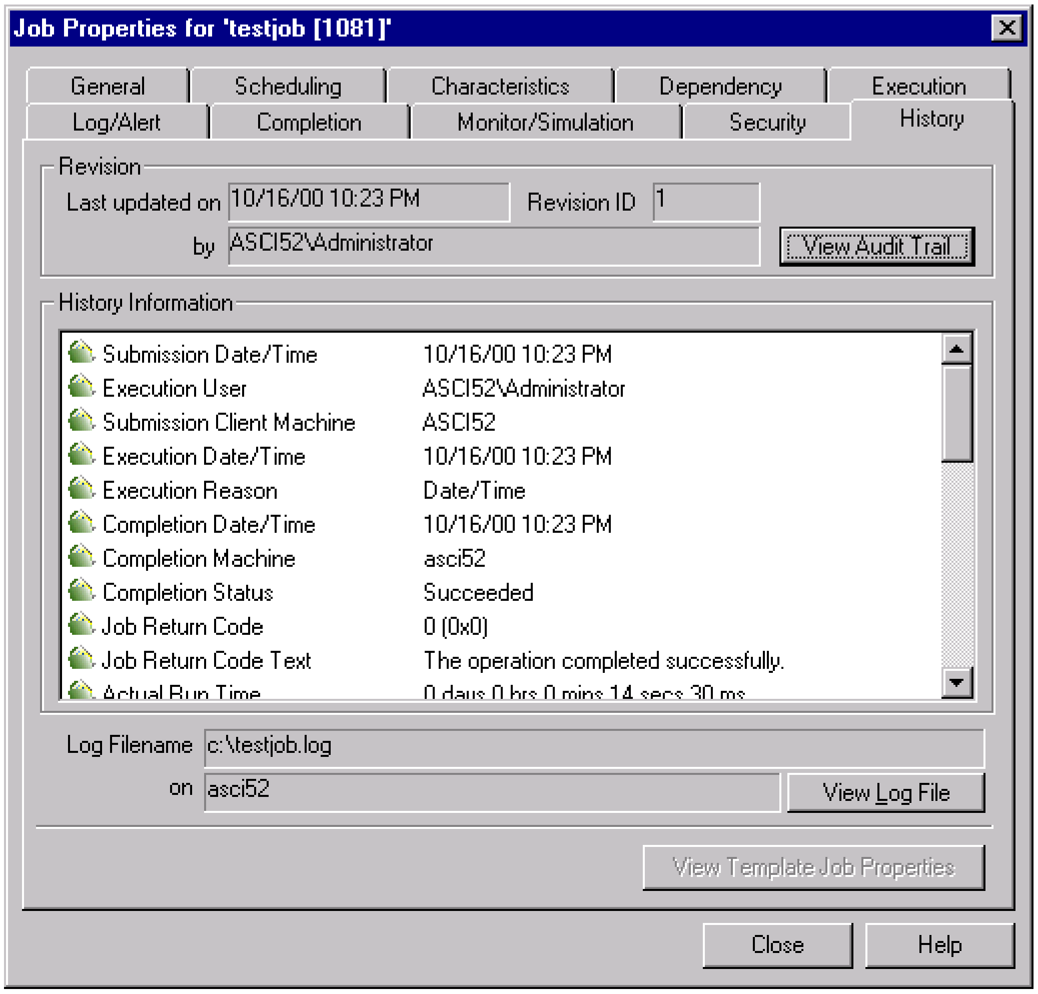
Submission Queue: quick_execq

Working Directory:

Queue Selection: Any Execution Manager ☐ Read Only

Figure 38 - Quick Tour Sample Job Tabs

Once a job has been submitted into the system, a "history" of the jobs interactions within ActiveBatch is created. ABATAdmin displays these properties under the "History" tab.



Revision

Last updated on 10/16/00 10:23 PM Revision ID 1

by ASCI52\Administrator

History Information

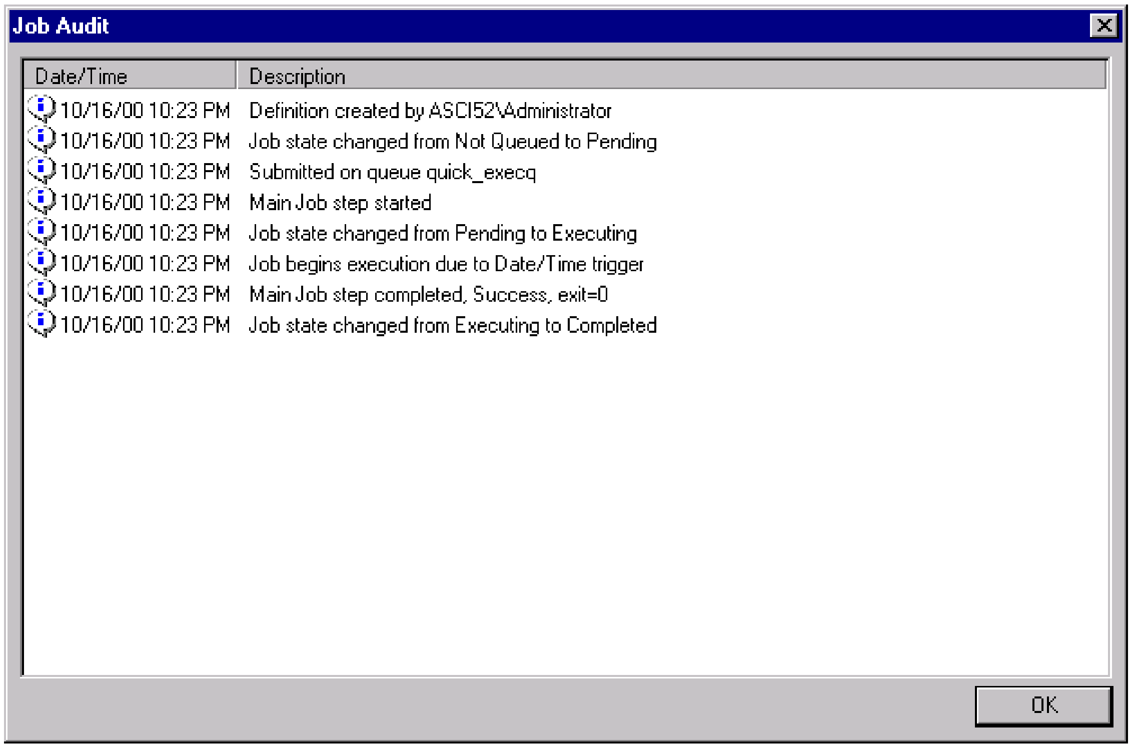
Submission Date/Time	10/16/00 10:23 PM
Execution User	ASCI52\Administrator
Submission Client Machine	ASCI52
Execution Date/Time	10/16/00 10:23 PM
Execution Reason	Date/Time
Completion Date/Time	10/16/00 10:23 PM
Completion Machine	asci52
Completion Status	Succeeded
Job Return Code	0 (0x0)
Job Return Code Text	The operation completed successfully.
Actual Run Time	0 days 0 hrs 0 mins 14 secs 30 ms

Log Filename c:\testjob.log

on asci52

Figure 39 - Quick Tour Job History Tab

Like any work-in-progress, the job's history becomes more complete as the job actually executes. Once the job instance has completely executed, a full display of the job's history is available. This includes all the various time and metrics concerning the job. One important aspect is the job's Audit Trail. This trail is the first place to look when a problem arises. The Audit Trail is a complete list of all the interactions between the job and ActiveBatch.



Date/Time	Description
10/16/00 10:23 PM	Definition created by ASCI52\Administrator
10/16/00 10:23 PM	Job state changed from Not Queued to Pending
10/16/00 10:23 PM	Submitted on queue quick_execq
10/16/00 10:23 PM	Main Job step started
10/16/00 10:23 PM	Job state changed from Pending to Executing
10/16/00 10:23 PM	Job begins execution due to Date/Time trigger
10/16/00 10:23 PM	Main Job step completed, Success, exit=0
10/16/00 10:23 PM	Job state changed from Executing to Completed

Figure 40 - Quick Tour Job Audit Trail

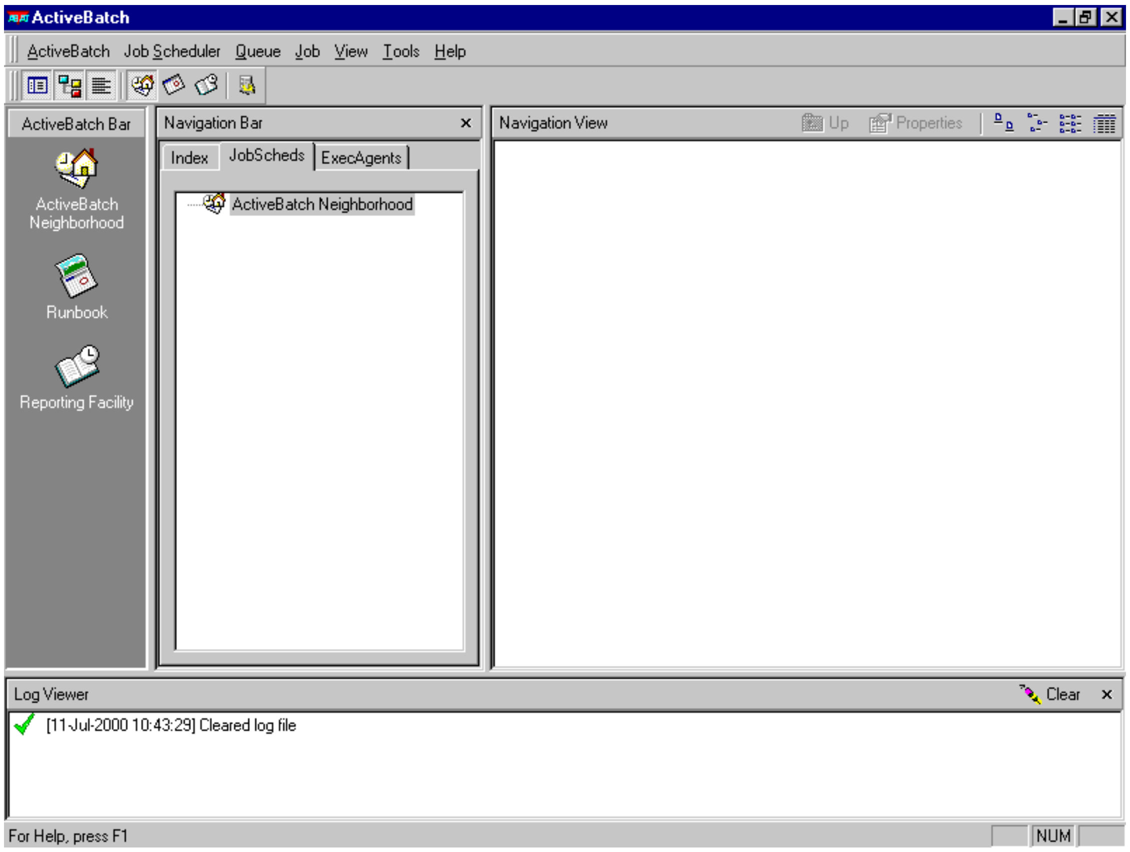
By examining the "Job History" information you can determine exactly what happened to your job.

6 USING ACTIVEBATCH WITH ABATADMIN

Hopefully you spent some time taking our Quick Tour (Section 5) and have become familiar with the system. This chapter will completely describe all the aspects of the ActiveBatch system in terms of usage. If you have questions concerning installation, as in what product edition to install or what component(s) can be installed please read Section 2.3. If you have questions concerning product startup, please read Section 3.3.

6.1 ABATAdmin Terms and Concepts

The ABATAdmin application is the user's main interface point into ActiveBatch. You can start ABATAdmin by either clicking on it's desktop icon or by selecting the program through the ActiveBatch program menu. ActiveBatch begins by displaying a "splash" screen identifying its version and registered owner. ABATAdmin continues the initialization process by connecting to all Job Schedulers you were previously connected to. ABATAdmin follows the general rule that whatever you were looking at on program exit you should be looking at when the program comes up. If this is the first time you're running the program, a screen similar to the one below will appear.



```
[11-Jul-2000 10:43:29] Cleared log file
```

Figure 41 - ABATAdmin Initial Usage

6.1.1 Bars and Views

Figure 41 shows the initial ABATAdmin screen with no connected Job Schedulers. This is the best time to explain the different *bars* and *views* that are presented. Like any Windows GUI program, ABATAdmin has a menu bar. The menu bar consists of the following menus: ActiveBatch, Job Scheduler, Queue, Job, View, Tools and Help. The menu bar is one method for specifying operations (like Connect to a Job Scheduler). Beneath the menu bar is the tool bar. The tool bar is a type of shortcut that allows you to quickly select one of the different views that ABATAdmin is capable of displaying. Simply place the cursor over a tool for a quick reminder as to what the tool provides.

The windows beneath the menu and tool bars vary based on the selected views. Three (3) views are currently available: ActiveBatch Neighborhood, Runbook and Reporting Facility. Selection of any of these views will alter the main windows accordingly. Each view presents ActiveBatch data differently to more appropriately match what you are trying to achieve. The ActiveBatch Neighborhood view is typically used when you are manipulating Queues and Jobs. The Runbook view is typically used when you are viewing completed jobs or manipulating jobs, especially scheduled jobs. The Reporting Facility isn't so much a view as a way of reporting about the jobs within the system.

The window at the bottom is the "Log Viewer" window and contains all informational, success and error messages concerning your use of ActiveBatch. A button to the far right of the window allows you to clear the log.

6.1.2 ActiveBatch Neighborhood View

The ActiveBatch Neighborhood view is the typical view used when manipulating ActiveBatch objects, especially Queue objects. This view is further divided into two (2) sub-windows: Navigation Bar and Navigation View.

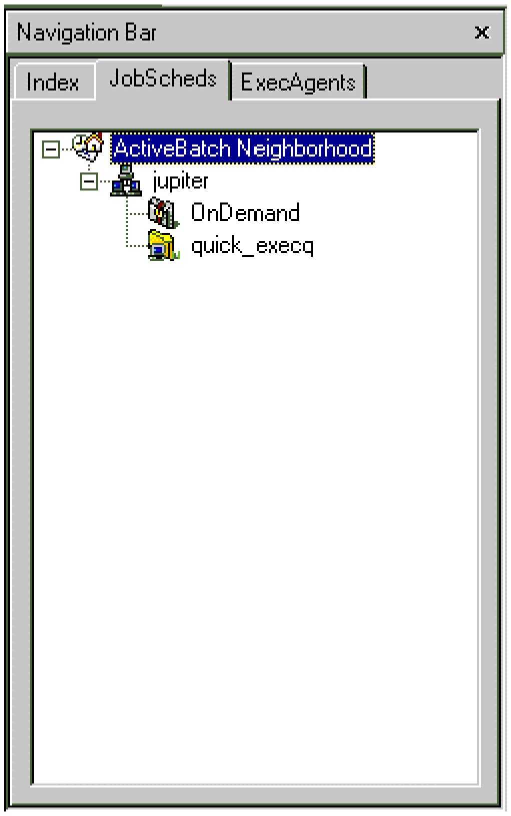


Figure 42 - Navigation Bar – Job Scheduler Tab

The Navigation Bar provides a three (3) tabbed view of the ActiveBatch Job Scheduler and Queue objects arranged depending on the tab chosen. The default tab is "Job Scheduler" and presents a tree oriented view of the ActiveBatch Neighborhood. In this view, all connected Job Scheduler machines are displayed with Queues presented as "leafs" within each Job Scheduler machine. This view is particularly useful since the objects are displayed in hierarchical order. A plus-sign (+) indicates that the object can be further exposed to display additional sub-objects. A minus-sign (-) indicates that the object has been fully exposed. Job Schedulers contain Queues (Generic, Execution and OnDemand) and Generic Queues contain Execution Queues. To perform operations on an object, simply select the object (item) and right click to display an in-context menu of the legal operations that can be performed.

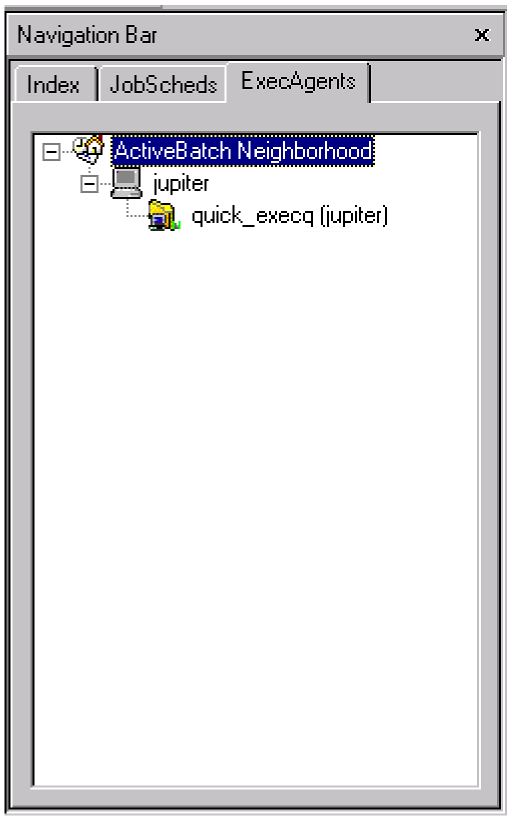
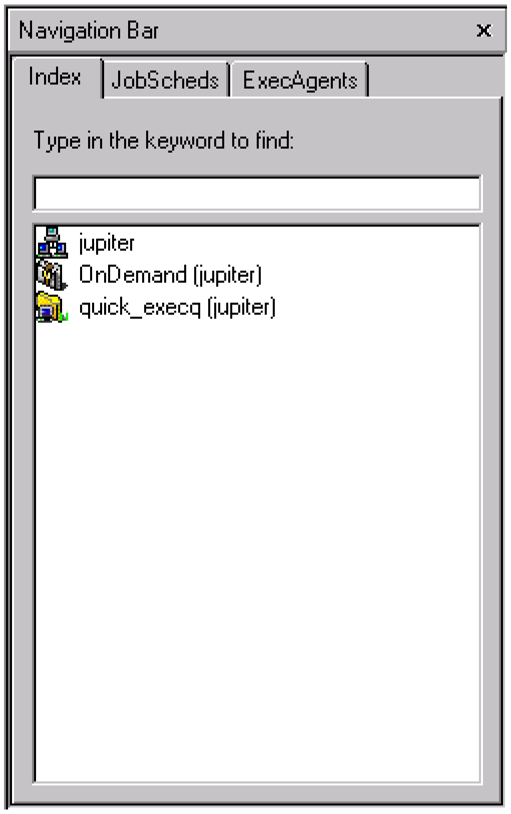


Figure 43 - Navigation Bar – Execution Agents Tab

This tab view displays only the Execution Queues and their execution machines within each Job Scheduler machine. This view is useful when you want to quickly identify your execution machines.



Type in the keyword to find:

jupiter
OnDemand (jupiter)
quick_execq (jupiter)

Figure 44 - Navigation Bar – Index Tab

This tab view displays all of the Job Scheduler, Queue and Job objects in alphabetical order. This view is very useful when you have lots of possible objects and you need to search for a specific object through its name. Simply typing a portion of the name causes that matching portion of the object(s) to be displayed. The parenthesis surrounding a queue indicates the Job Scheduler name. The parenthesis surrounding a job indicates the Job Scheduler and Queue.

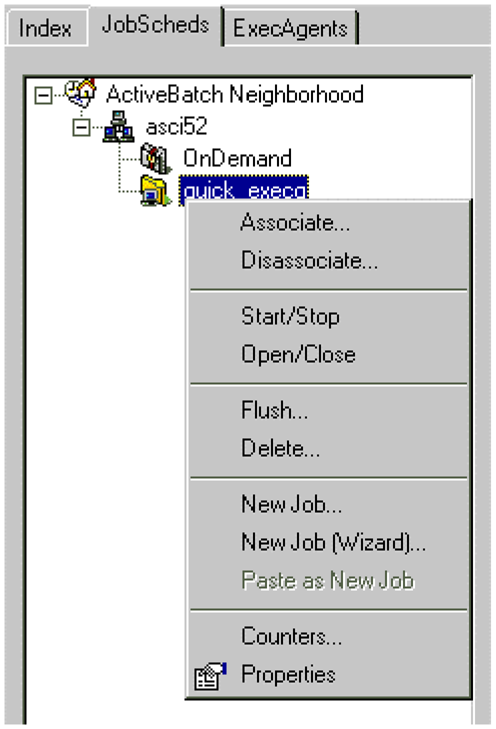
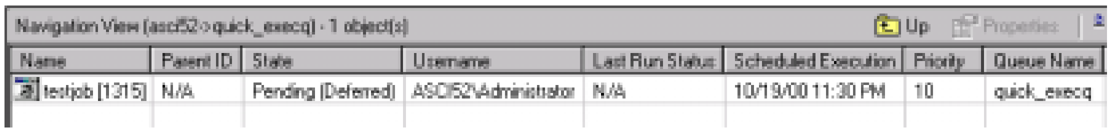


Figure 45 - Selection and Right-Click Menu

Regardless of the tab view selected, ActiveBatch menus are always in-context. This means that certain selections will be grayed-out if the selection isn't appropriate. For example, if you have selected a Queue item, the Job menu operations aren't made available. Finally, you can remove the Navigation Bar window by clicking on the "x" in the right hand corner of the bar window.

The Navigation View displays any related sub-items or objects. The Navigation View bar displays the current item selected in the Navigation Bar and allows further refinement of the presented view. By default, these objects are displayed in Details mode. You can select to display small or large icons or a simple list (just like Explorer) in the Navigation View by pressing the appropriate button on the Navigation View bar. You can also move one level up by pressing the "Up" button.



Name	Parent ID	State	Username	Last Run Status	Scheduled Execution	Priority	Queue Name
testjob [1315]	N/A	Pending (Deferred)	ASCI52\Administrator	N/A	10/19/00 11:30 PM	10	quick_execq

Figure 46 - Navigation View (Details)

6.1.3 ActiveBatch Icons

As you look through the various ABATAdmin views you've probably noticed several icons. ActiveBatch makes very liberal use of icons and this section describes each icon you will see in the system. ActiveBatch icons do not use color to denote object type. So even on a black and white monitor you will be able to differentiate between a queue and a job. The icons do use color to denote state information.

Queues are represented as a folder icon similar to a directory. If the folder is open, the queue is open. Use of color denotes the state. Green means the queue is started. Red means the queue is starting or stopped. If the folder is open, the queue is open. If the folder is closed, the queue is closed.

An Execution Queue contains a small machine next to the folder. A Generic Queue contains a series of smaller folders next to the larger one.

The following icons represent various queues in various states. As you can imagine there are many variations for each object so to keep the information concise and not confusing the four (4) start states will be shown for an open queue. The state portion of the icon is the same for the queue when it is in a closed state.

This is an Execution Queue that is both open and started.

This is an Execution Queue that is open but starting (which essentially means that it's currently stopped and waiting for the Execution Agent to respond).

This is an Execution Queue that is open but stopping.

This is an Execution Queue that is open but stopped.

This is an Execution Queue that is closed and started.

This is a Generic Queue that is open and started.

This is an On-Demand Queue.

This is an Execution Queue in an unknown state. Obviously this shouldn't happen.

The next set of icons concern Jobs. ActiveBatch jobs come in two major categories: Templates and Instances. Templates are always active jobs in the sense that a template waits for some triggering action or event to run a job instance. Templates are depicted as a series of job icons (similar to the series of small folders for a generic queue).

This is a template job. A template job's default state is pending.

This is a job that is currently executing.

This is a job that is currently pending.

This is a job that is held.

This is a job that has completed successfully.

This is a job that has completed in failure.

This is a job that has been aborted.

This job has been paused (was in executing state).

This is a job in an unknown state. Obviously this should not happen.

These icons are used by the JobScheds and Exec Agents tab in the Navigation Bar to denote additional objects.

This is an Execution machine.

This is a Job Scheduler machine that doesn't support AutoRefresh.

This is a Job Scheduler machine with AutoRefresh.

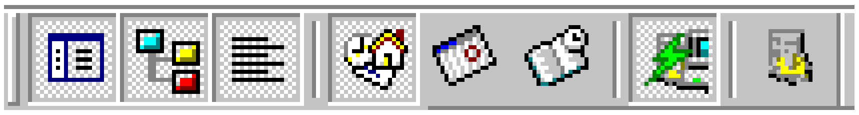


Figure 47 - Tool Bar

The Tool Bar (Figure 47) located under the menu bar is a quick way of changing various views and options. From left to right:

- ActiveBatch Bar. This bar allows you to quickly select from the three (3) possible views.
- Navigation Bar. This bar provides the queue system in a tree-view.
- Log View. This bar controls the Logger View of ActiveBatch messages.
- Navigation View. This view provides an iconized or list of queues and their jobs.
- Runbook View. This diary view provides access to past, present and future jobs.
- Reporting View. This view provides various reports of past jobs.
- Automatic WMI Refresh. WMI enabled for possible automatic object/state refresh.
- Options.

The Tool buttons act as real buttons. Depressing the button enables the feature or view.

6.1.4 ActiveBatch Hot-Keys

ABATAdmin supports the use of “hot-keys” for those advanced users who would prefer to use the keyboard for faster product interaction than your mouse. The following keys and operations are supported. Please note that a single key will perform more than one action. The action depends entirely on the current selected context. The **F1** key may be used at anytime to request on-line help and the **F5** key causes a manual data refresh.

Job Scheduler

F4	Connect to Job Scheduler
F10	Properties
F11	Schedules
F12	Calendars

Queue

F10	Properties

Job

F2	New
F3	New (Wizard)
F10	Properties

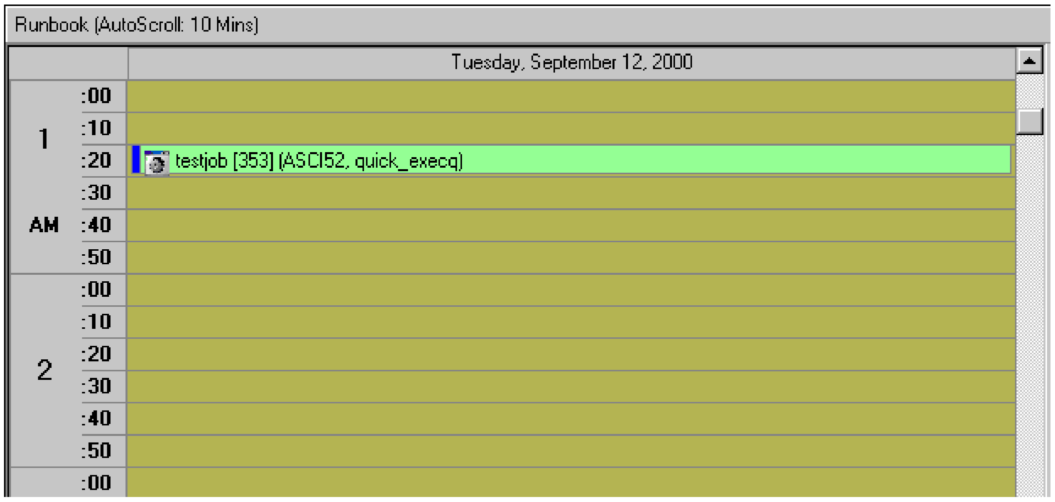
Views

F6	Navigation
F7	Reporting
F8	Runbook

6.1.5 Runbook View

The Runbook View uses a diary orientation to display jobs as an operator would want to see the work as it is scheduled throughout the day, week or month. In the Runbook view, jobs are presented in date and time sequence. The Runbook view is the only view in ActiveBatch in which you can access past, present and future jobs.

Selecting the Runbook View causes three (3) windows to be displayed. The main window is the runbook itself which is presented in a diary type format. The number of days displayed depends on the current selection. You may select 1 day, 5 day or 7 day runbook views. Obviously, the 1 day view can display the most detailed information. In the 1-day view, the time is listed along the left hand portion of the window and a vertical scrollbar is available for scrolling through the day.



testjob [353] (ASCI52, quick_execq)

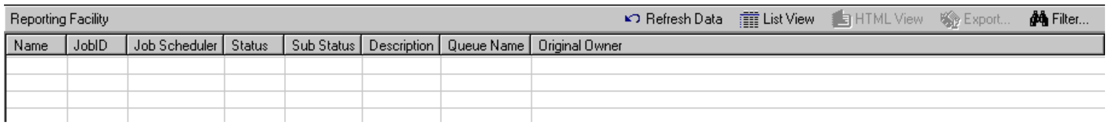
Figure 48 - Runbook Main Window (1 Day)

Each vertical bar in the above figure represents ten (10) minutes. You can select intervals from five minutes to one hour (see Section 6.1.7). Past, present and future jobs are displayed at the appropriate times. Jobs are displayed using various colors. **Red** indicates a past job that has failed. **Green** indicates a past job that has succeeded. **Grey** indicates a past job that has been aborted. **White** indicates a presently executing job and **Yellow** indicates a future job instance. The right hand side of the window consists of a monthly calendar in the upper portion and a job properties or selection sheet in the lower portion. If the time period contains more than three (3) jobs, clicking on that time period causes the job selection window to display all the jobs which can be further selected. Double-clicking on a job in the job selection window or double-clicking on a job in the main window causes the Job Properties tabs to be displayed.

The monthly calendar portion of the window allows you to select a day or period to be displayed. The actual day or period displayed is determined by the day specification (1 day, 5 day, 7 day). Jobs which are scheduled to execute on a certain day are displayed by the day appearing in italics.

6.1.6 Reporting Facility

The Reporting Facility allows you to extract and retrieve information about ActiveBatch jobs.



Name	JobID	Job Scheduler	Status	Sub Status	Description	Queue Name	Original Owner

Figure 49 - Reporting Facility

The Reporting Facility allows you to report on ActiveBatch jobs through filtering criteria. This means you can look at the past or present system and produce a report that meets your needs. If you examine the Reporting toolbar you will see several icons which allow you to customize and create the report.

Refresh Data – Resets the view.

List View – This is the default view and provides a typical list view of the report elements.

HTML View – This view causes the report to be generated using HTML. You will want to specify this type of view to print out the report.

Export – This button lets you export your report into a Comma Separated file that can be re-imported into most databases and spreadsheets.

Filter – This button controls what jobs are selected for inclusion into the report.

6.1.6.1 Creating a Report

You create a report by indicating one or more filters that are used to select the jobs you want to report on.

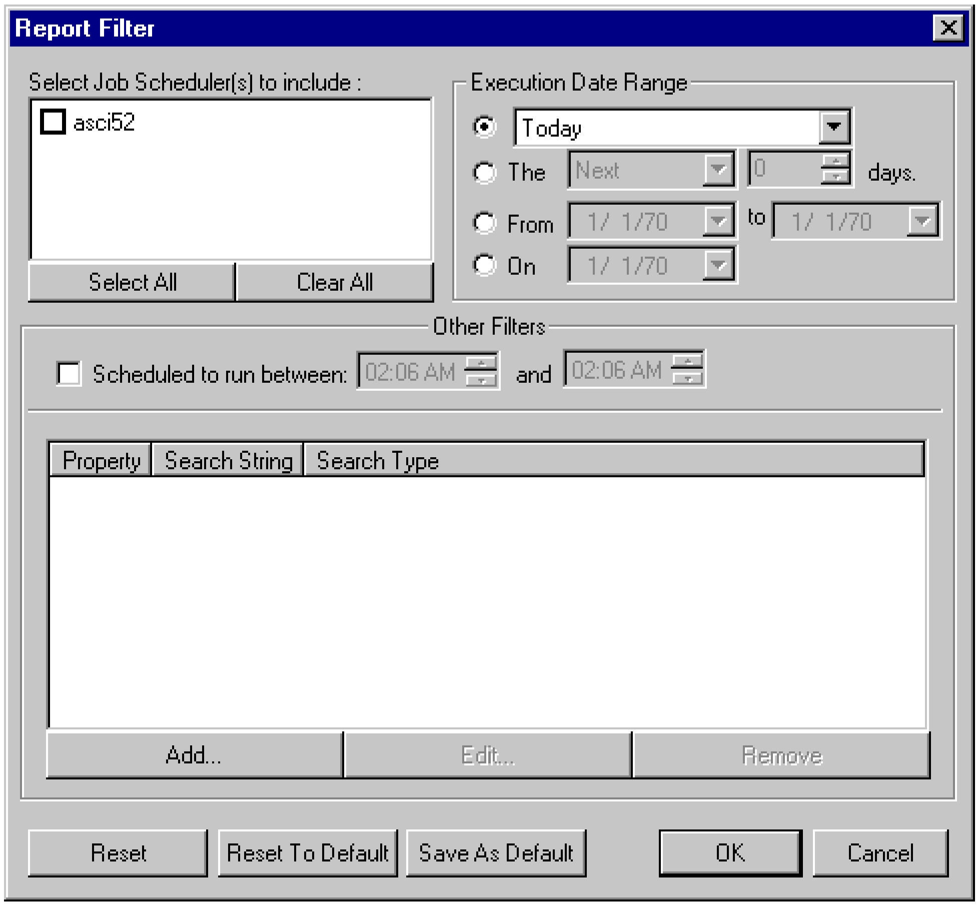
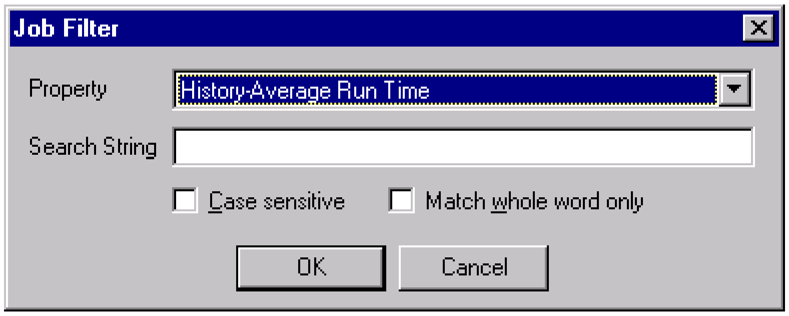


Figure 50 - Report Filter Dialog

The filters shown in Figure 50 fall into three (3) categories. Job Scheduler, Date and Content filters. The Job Scheduler section is simple. Click on each currently connected Job Scheduler to include its database within the Date and Content filters that may be specified. The Date filter provides four (4) mutually exclusive choices: Current Week/Today, the last/next *n* days, a *from* and *to* range and a job executed date. You may also specify a time range to further refine the list of jobs produced. The Content filter allows you to filter based on any Job property. The choices are additive so you can further refine the report. To add a content filter, click on the *Add* button and specify the property, search string and type of search (case sensitive and/or match whole word only). Once you have made your selection, you can press *OK* to confirm and process the report or *Cancel* to abort the filter selection. Three convenient buttons are also provided along the bottom. *Reset* allows you to quickly clear the dialog box. *Reset To Default* let's you clear the dialog box back to its default settings. *Save As Default* let's you save your filter settings so you can produce the same report or type of report quickly without re-entering the data.



Property: History-Average Run Time

Figure 51 - Report Filter Add Dialog

6.1.6.2 Exporting Report Information

Once you've produced your report you may want to export the information into a spreadsheet or other database so you can perform any custom analysis of the information you need. The COM aspects of ActiveBatch allow you to create macros which can directly access any ActiveBatch object and property. The *Export* button allows you to accomplish some of this functionality in a simple manner. When you press that button you'll be presented with a dialog box in which you can specify the location and name of the extract file to be created. The extract file is a Comma Separated file (.CSV). This type of file is most commonly used when importing data into a spreadsheet or database.

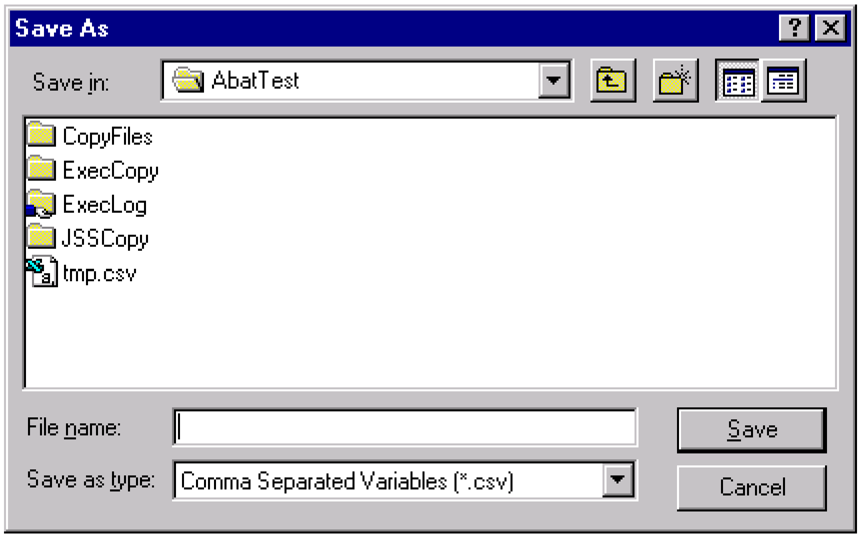
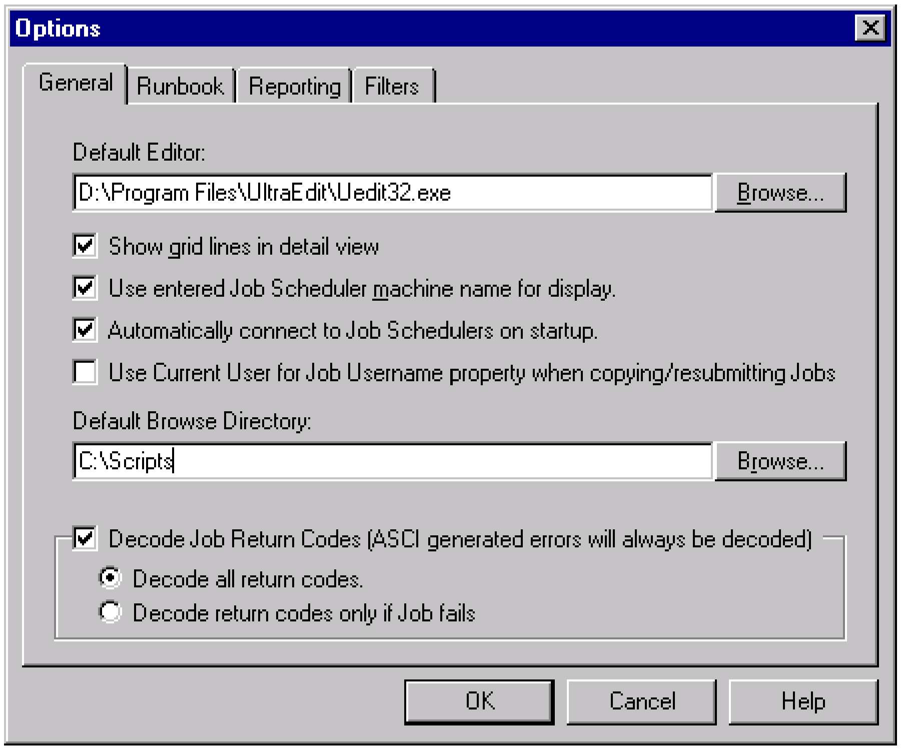


Figure 52 - Exporting Data

6.1.7 Tools and Options

The *Tools* menu option allows you to select whether the program should save its layout changes on exit or immediately. Tools also contains the *Options* sub-menu which allows you to specify your preferences. Four (4) tabs are provided for setting various preferences depending on program area.



Default Editor: D:\Program Files\UltraEdit\Uedit32.exe

[✓] Show grid lines in detail view

[✓] Use entered Job Scheduler machine name for display.

[✓] Automatically connect to Job Schedulers on startup.

[ ] Use Current User for Job Username property when copying/resubmitting Jobs

Default Browse Directory: C:\Scripts

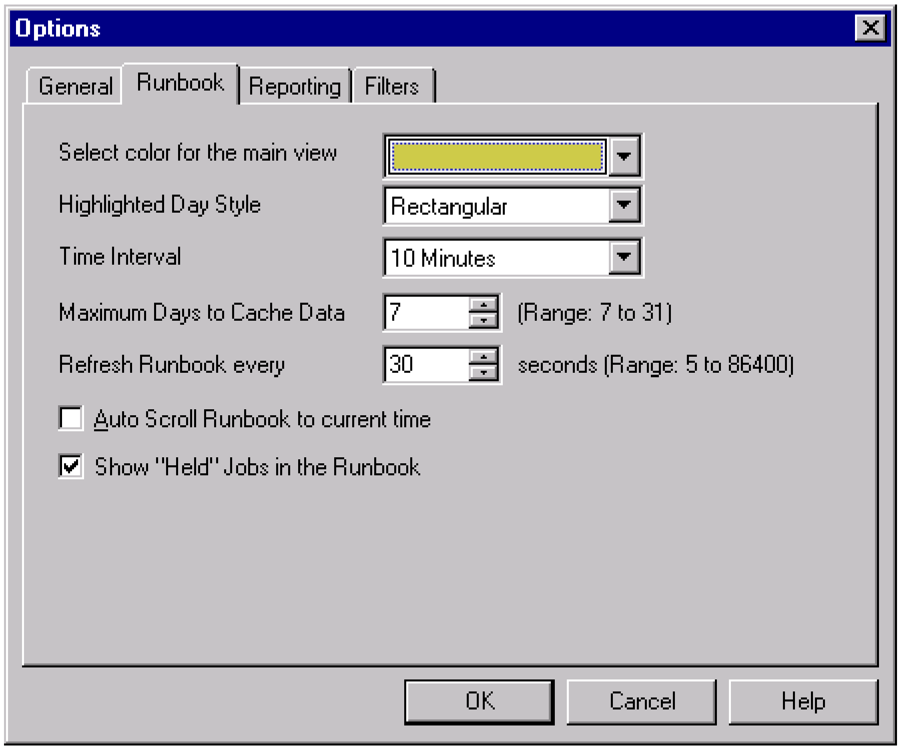
[✓] Decode Job Return Codes (ASCI generated errors will always be decoded)

● Decode all return codes.

○ Decode return codes only if Job fails

Figure 53 - Options (General)

This page allows you to designate an editor to be used when viewing a job's log file. The first three options are self-explanatory. *Use Current User…* controls whether your currently logged in username should be inserted on a Copy or Resubmit of a job. Cross Platform users, in particular, will want to keep this checkbox cleared. You can specify the default directory to be used during a *Browse* sequence. You can also indicate whether you want your Job's exit code to be interpreted and, if so, under what conditions.



Select color for the main view: 🟨

Highlighted Day Style: Rectangular

Time Interval: 10 Minutes

Maximum Days to Cache Data: 7 (Range: 7 to 31)

Refresh Runbook every: 30 seconds (Range: 5 to 86400)

☐ Auto Scroll Runbook to current time

☑ Show "Held" Jobs in the Runbook

Figure 54 - Options (Runbook)

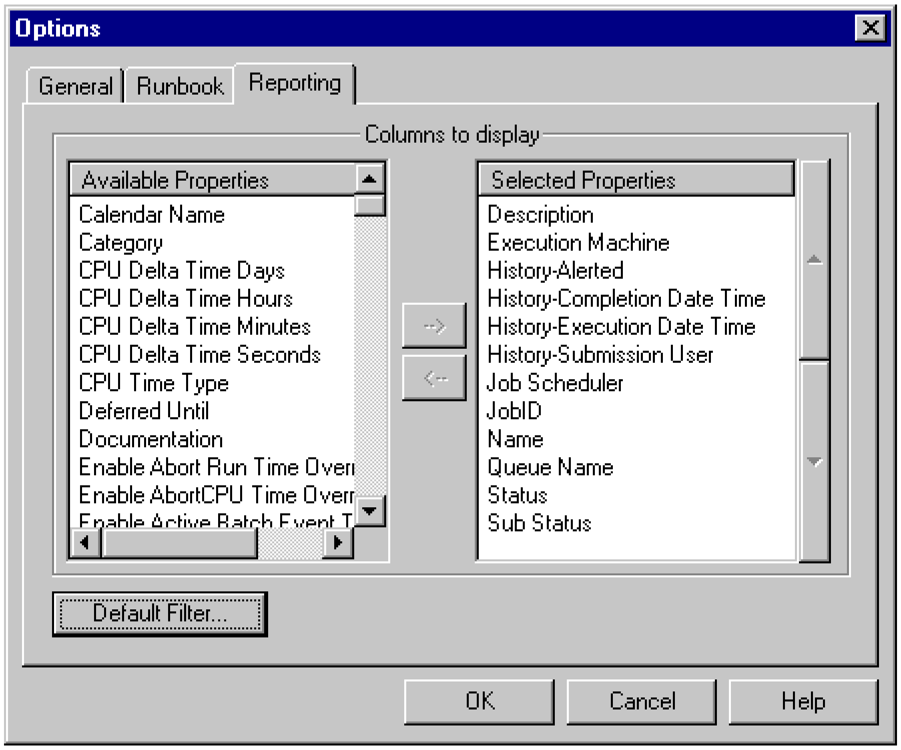
This tab allows you to specify preferences when using the Runbook view. The last three items bear some discussion.

Time Interval. This option allows you to set the default time interval used when displaying the "diary" portion of the Runbook view. You may specify an interval from five (5) minutes to one (1) hour. The initial default is ten (10) minutes.

Maximum Days to Cache Data. This option sets the maximum number of Runbook days that will be cached in memory. The legal range is 7 to 31 days. The default is 7 days.

Auto Scroll Runbook to current time. This checkbox (enabled by default) will cause the diary portion when viewing the current day to automatically scroll based on time of day. This is very useful when you want to follow along with jobs that are scheduled to run.

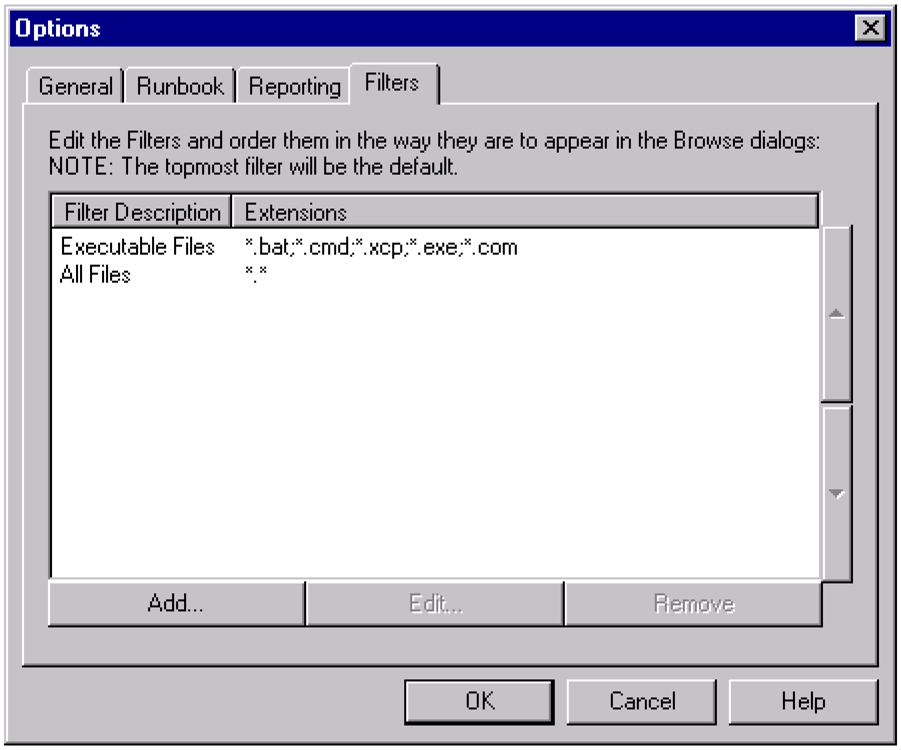
Show "Held" Jobs in the Runbook. This checkbox (enabled by default) will allow held jobs to be displayed In the Runbook. You might want to disable this if you hold many template jobs.



Available Properties	Selected Properties
Calendar Name	Description
Category	Execution Machine
CPU Delta Time Days	History-Alerted
CPU Delta Time Hours	History-Completion Date Time
CPU Delta Time Minutes	History-Execution Date Time
CPU Delta Time Seconds	History-Submission User
CPU Time Type	Job Scheduler
Deferred Until	JobID
Documentation	Name
Enable Abort Run Time Overr	Queue Name
Enable AbortCPU Time Overr	Status
Enable Active Batch Event T	Sub Status

Figure 55 - Options (Reporting)

This tab allows you to set the job properties that you would like to display when viewing a report. The number of job properties is too large to discuss here although every property is discussed in the *ActiveBatch Developer's Guide*. The area labeled “Available Properties” is what you choose from using the LEFT or RIGHT arrow buttons to add or remove those properties to/from the “Selected Properties”. The two major UP and DOWN arrow buttons allow you to change the column order of those selected properties. The button labeled *Default Filter* allows you to create and save a filter that will be used by default when creating a report.



Edit the Filters and order them in the way they are to appear in the Browse dialogs:
NOTE: The topmost filter will be the default.

Filter Description	Extensions
Executable Files	*.bat;*.cmd;*.xcp;*.exe;*.com
All Files	*.*

Figure 56 - Options (Filters)

This tab controls which file extensions are displayed during a *Browse* sequence. The example above indicates that those files with extensions of .bat, .cmd, .xcp, .exe and .com will be displayed. You may enter as many file filters as you desire. The topmost filter will always be used when you initially enter a *Browse* sequence. The Add, Edit and Remove buttons allow you to add, edit and remove filter rules. The Up and Down Arrow buttons on the right allow you to change the order of the filter rules as they are displayed.

6.1.8 ABATAdmin and WMI

ABATAdmin provides an automatic refresh of ActiveBatch objects through its support of Microsoft's Windows Management Instrumentation (WMI) facility. The automatic refresh mechanism requires that the Client machine as well as the Job Scheduler machine have WMI installed and started. If WMI is not present or started on either machine, a manual refresh of ActiveBatch data will be necessary to see any changes which have occurred (you can press the F5 key to request a manual refresh). With WMI enabled, all object changes are sent to ABATAdmin as events. Each event is specific to the object that changed. By using WMI instead of some "homegrown" method, the publishing of event data is made available both locally and remote through a facility which can be used by anyone.

ABATAdmin indicates that automatic refresh (AutoRefresh) is available on a Job Scheduler machine through the use of a special icon. The "lightening bolt" indicates that automatic refresh is available. ABATAdmin provides automatic refresh for the Navigation and Runbook views.

6.2 Job Scheduler

6.2.1 Connecting to a Job Scheduler

With ABATAdmin running you're ready to connect to a Job Scheduler. The Job Scheduler component is an NT Service that manages all the ActiveBatch objects. ActiveBatch supports one or more Job Schedulers. As the name implies, this service handles job and machine scheduling. So to do something in ActiveBatch requires that you first connect to a Job Scheduler.

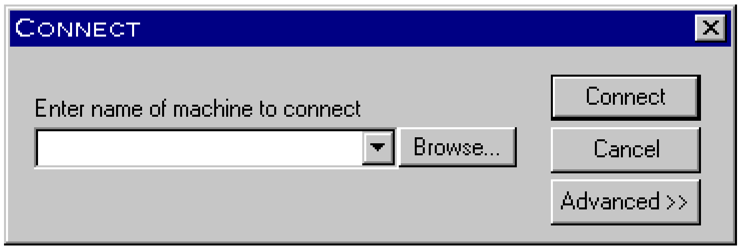


Figure 57 - Connect Dialog (Basic)

To connect to a Job Scheduler machine, either right click on the "ActiveBatch Neighborhood" icon and select "Connect…" or click on "Job Scheduler/Connect…" using the menu bar. The figure above shows the Connect dialog. The Basic Connect dialog simply asks for a machine name (or IP address) to connect to. You may enter the machine name by typing it in, or selecting a previously entered machine from the recent drop down list, or by clicking on the *Browse* button to display a Network Neighborhood listing of machines. When you have selected a machine, click on the *Connect* button to attempt the connection. A successful connection results in a list of that Job Scheduler's Queues presented (and merged) into your current view.

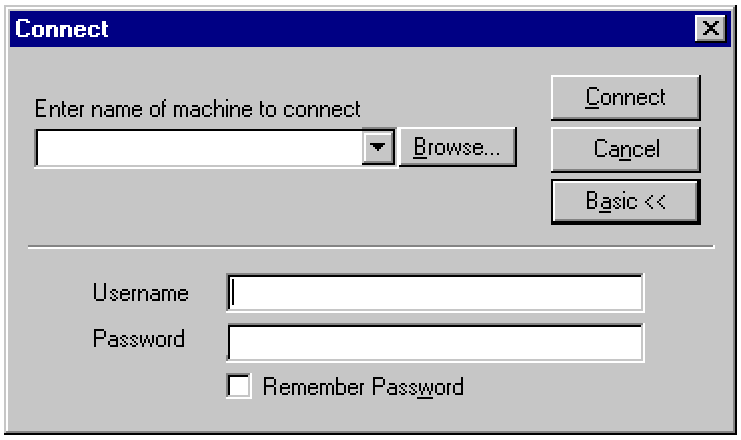


Figure 58 - Connect Dialog (Advanced)

The Basic Connect dialog assumes that your current authentication credentials are sufficient to connect to the named Job Scheduler machine. Typically, for most users, this will be the case. However, if you are logged onto a Windows 9x system or a Windows 2000/NT machine that is not part of the security domain of the Job Scheduler machine you will need to present your authentication credentials to ActiveBatch so we can authenticate you. To perform an extended authentication check, click the *Advanced*>> button. This will cause Username and Password fields to be displayed. You should enter a username and password that can be authenticated for the selected Job Scheduler system. This will typically be a domain user account for Windows 2000/NT. As a convenience you can click the checkbox for *Remember Password* to have ActiveBatch remember your entered username/password for the selected Job Scheduler machine. This saved information will only be used for establishing a connection to a Job Scheduler machine. Nothing more. Once you've entered the information you may click the *Connect* button to allow ActiveBatch to attempt a connection with the authentication credentials you've supplied.

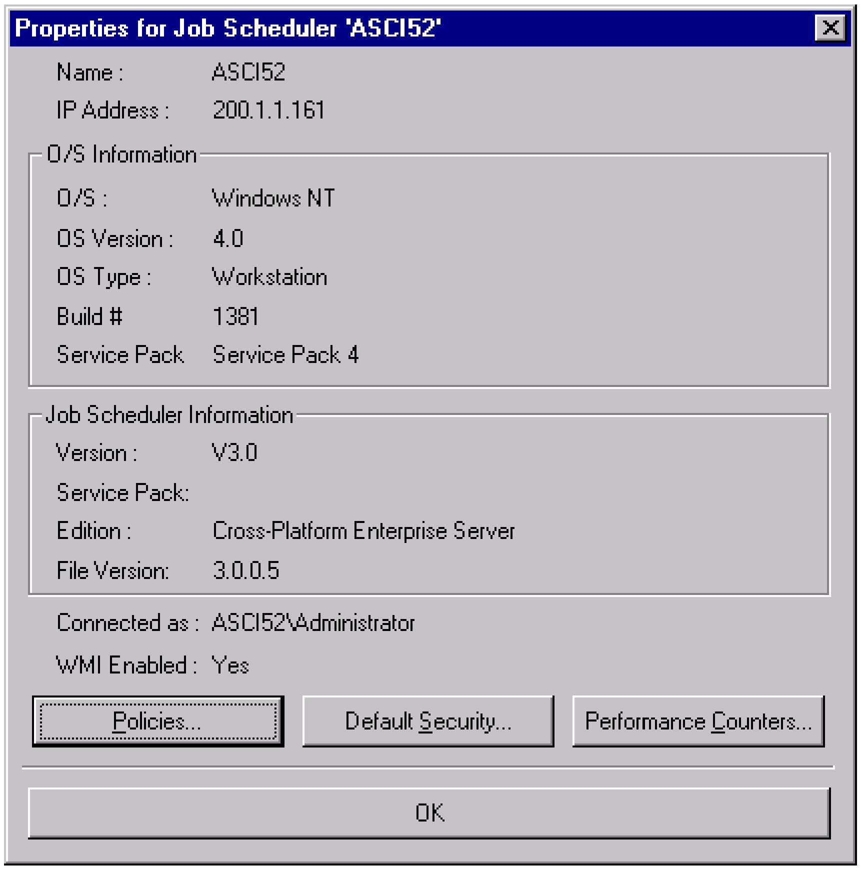
6.2.2 Disconnecting from a Job Scheduler

To disconnect from a Job Scheduler you can either right click on the selected Job Scheduler machine icon and select *Disconnect* or you can select the Job Scheduler machine icon and choose *Job Scheduler/Disconnect* from the menu bar. When you successfully disconnect from a Job Scheduler the entire tree of Queues that Job Scheduler represented are removed from view.

Note: You do not have to be connected to a Job Scheduler for your previously submitted jobs to properly execute.

6.2.3 Job Scheduler Properties

Clicking on the Job Scheduler icon and then selecting *Properties* provides the following dialog box:



Properties for Job Scheduler 'ASCI52'

Name :	ASCI52
IP Address :	200.1.1.161

O/S Information

O/S :	Windows NT
OS Version :	4.0
OS Type :	Workstation
Build #	1381
Service Pack	Service Pack 4

Job Scheduler Information

Version :	V3.0
Service Pack:	
Edition :	Cross-Platform Enterprise Server
File Version:	3.0.0.5

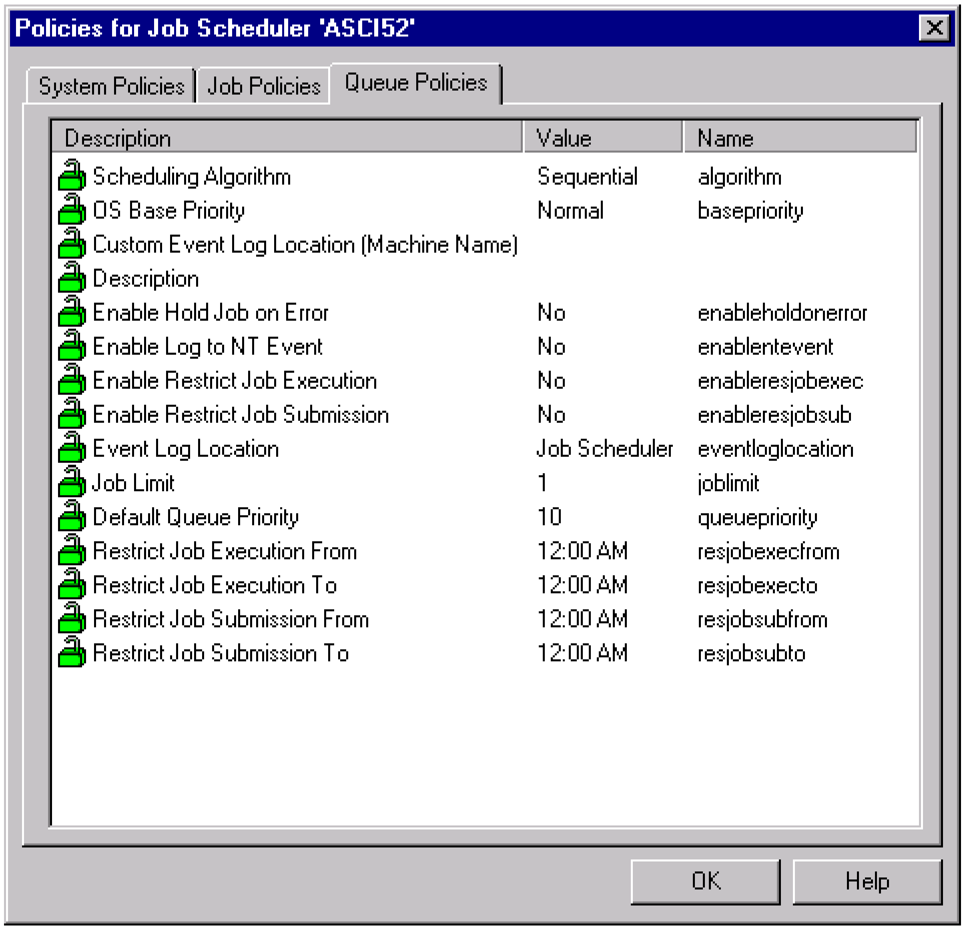
Connected as :	ASCI52\Administrator
WMI Enabled :	Yes

Figure 59 - Job Scheduler Properties

The information displayed above is retrieved from the Job Scheduler's machine. The *Connected As* information reminds you of the current security context associated with this Scheduler and the *WMI Enabled* property indicates whether the Job Scheduler (and machine) are using WMI Event handling (necessary for automatic refresh). You can view additional information on the Job Scheduler such as Policy Information, Default Security Permissions and Performance Counters.

You can only look at the information provided. To change this data you must use the ActiveBatch MMC snap-in (as described in Section 0).

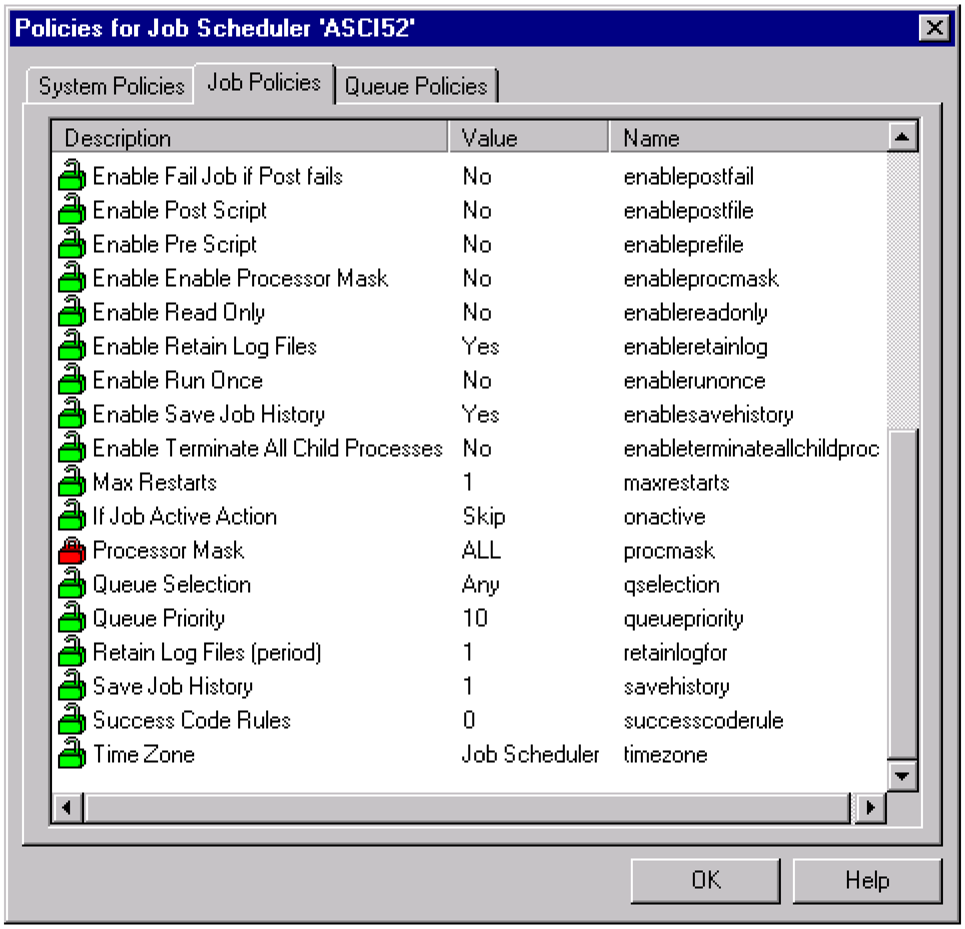
Policy Information is discussed in Section 8.4. Three (3) types of policies are viewable from the Job Scheduler's Properties dialog box: Queue, Job and System.



Description	Value	Name
Scheduling Algorithm	Sequential	algorithm
OS Base Priority	Normal	basepriority
Custom Event Log Location (Machine Name)		
Description		
Enable Hold Job on Error	No	enableholdonerror
Enable Log to NT Event	No	enablentevent
Enable Restrict Job Execution	No	enableresjobexec
Enable Restrict Job Submission	No	enableresjobsub
Event Log Location	Job Scheduler	eventloglocation
Job Limit	1	joblimit
Default Queue Priority	10	queuepriority
Restrict Job Execution From	12:00 AM	resjobexecfrom
Restrict Job Execution To	12:00 AM	resjobexecto
Restrict Job Submission From	12:00 AM	resjobsubfrom
Restrict Job Submission To	12:00 AM	resjobsubto

Figure 60 - Queue Policies (Job Scheduler)

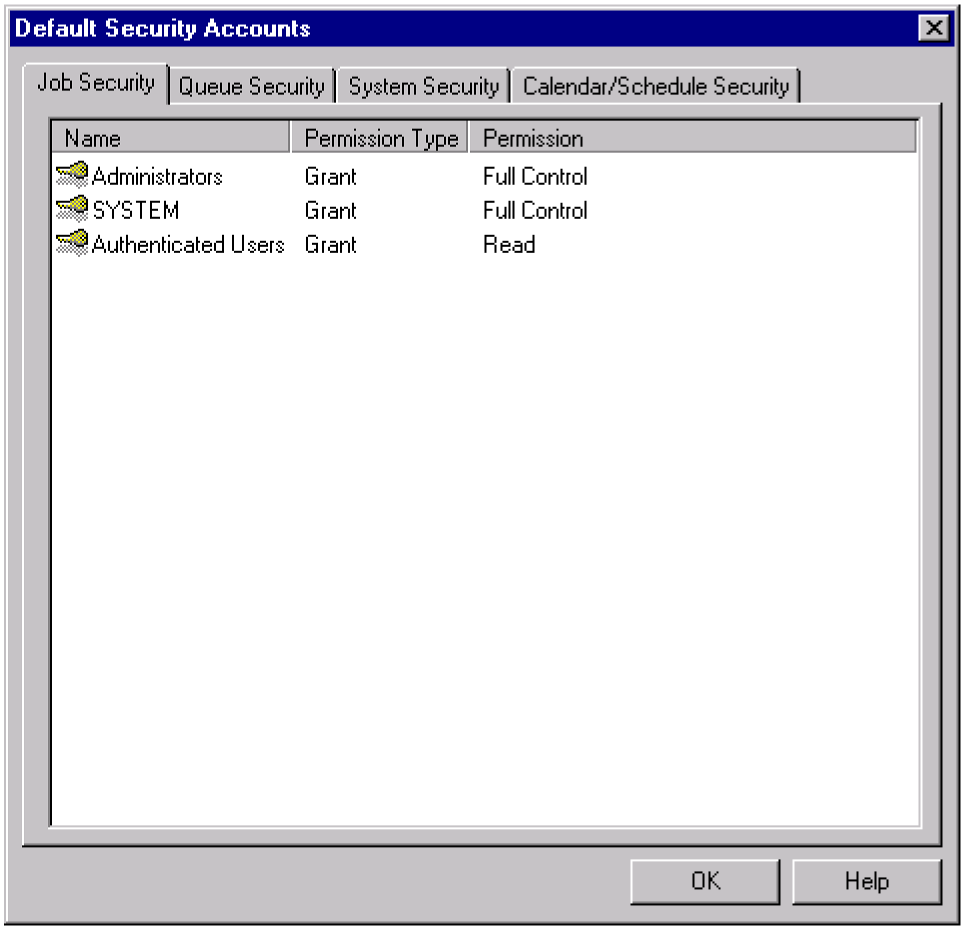
You'll notice little green locks next to the Queue Policy names. Those green locks mean that the default values you see next to most of the Queue properties can be changed on the various Queue property tabs. A red lock means that the policy value cannot be changed unless you're an ActiveBatch Administrator.



Description	Value	Name
Enable Fail Job if Post fails	No	enablepostfail
Enable Post Script	No	enablepostfile
Enable Pre Script	No	enableprefile
Enable Enable Processor Mask	No	enableprocmask
Enable Read Only	No	enablereadonly
Enable Retain Log Files	Yes	enableretainlog
Enable Run Once	No	enablerunonce
Enable Save Job History	Yes	enablesavehistory
Enable Terminate All Child Processes	No	enableterminateallchildproc
Max Restarts	1	maxrestarts
If Job Active Action	Skip	onactive
Processor Mask	ALL	procmask
Queue Selection	Any	qselection
Queue Priority	10	queuepriority
Retain Log Files (period)	1	retainlogfor
Save Job History	1	savehistory
Success Code Rules	0	successcoderule
Time Zone	Job Scheduler	timezone

Figure 61 - Job Policies (Job Scheduler)

You'll notice on Figure 61 that one of the policies (Processor Mask) has a little red lock. That means that value pertaining to that Job Policy cannot be changed unless you're an ActiveBatch Administrator.

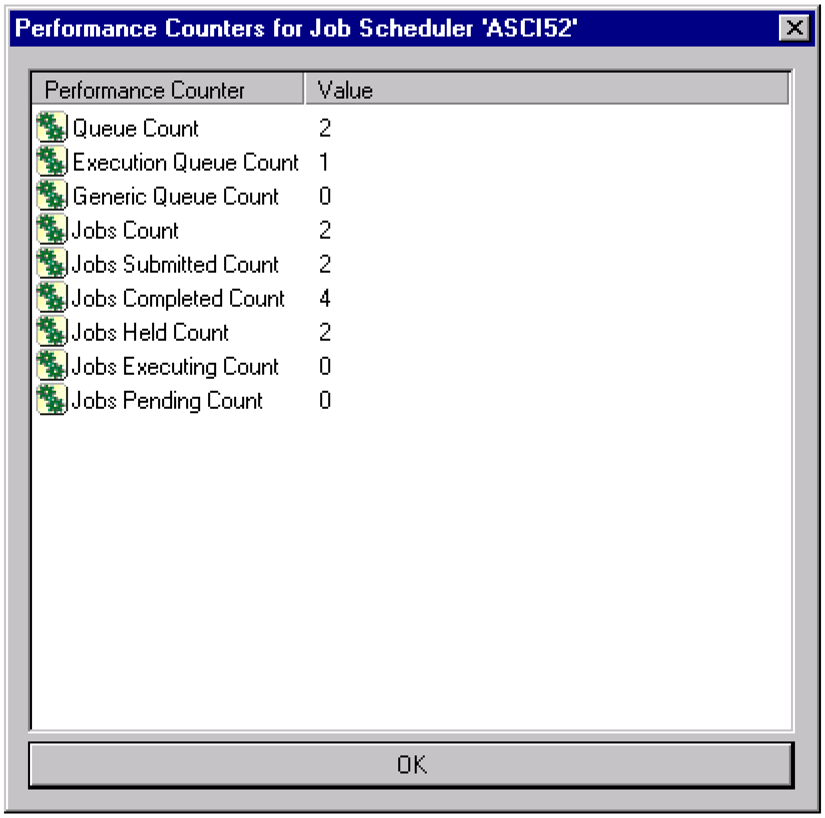


Name	Permission Type	Permission
Administrators	Grant	Full Control
SYSTEM	Grant	Full Control
Authenticated Users	Grant	Read

Figure 62 - Job Scheduler Default Security

The figure above shows the default security access for each of the ActiveBatch objects that are secureable.

The figure below shows a sample of Job Scheduler Performance counters.



Performance Counter	Value
Queue Count	2
Execution Queue Count	1
Generic Queue Count	0
Jobs Count	2
Jobs Submitted Count	2
Jobs Completed Count	4
Jobs Held Count	2
Jobs Executing Count	0
Jobs Pending Count	0

Figure 63 - Job Scheduler Performance Counters

The counters in this list box are of the "snapshot" variety. They won't change unless you refresh the box. The good news is that all the performance counters are exported so you can track historical and operational counts through various Windows facilities.

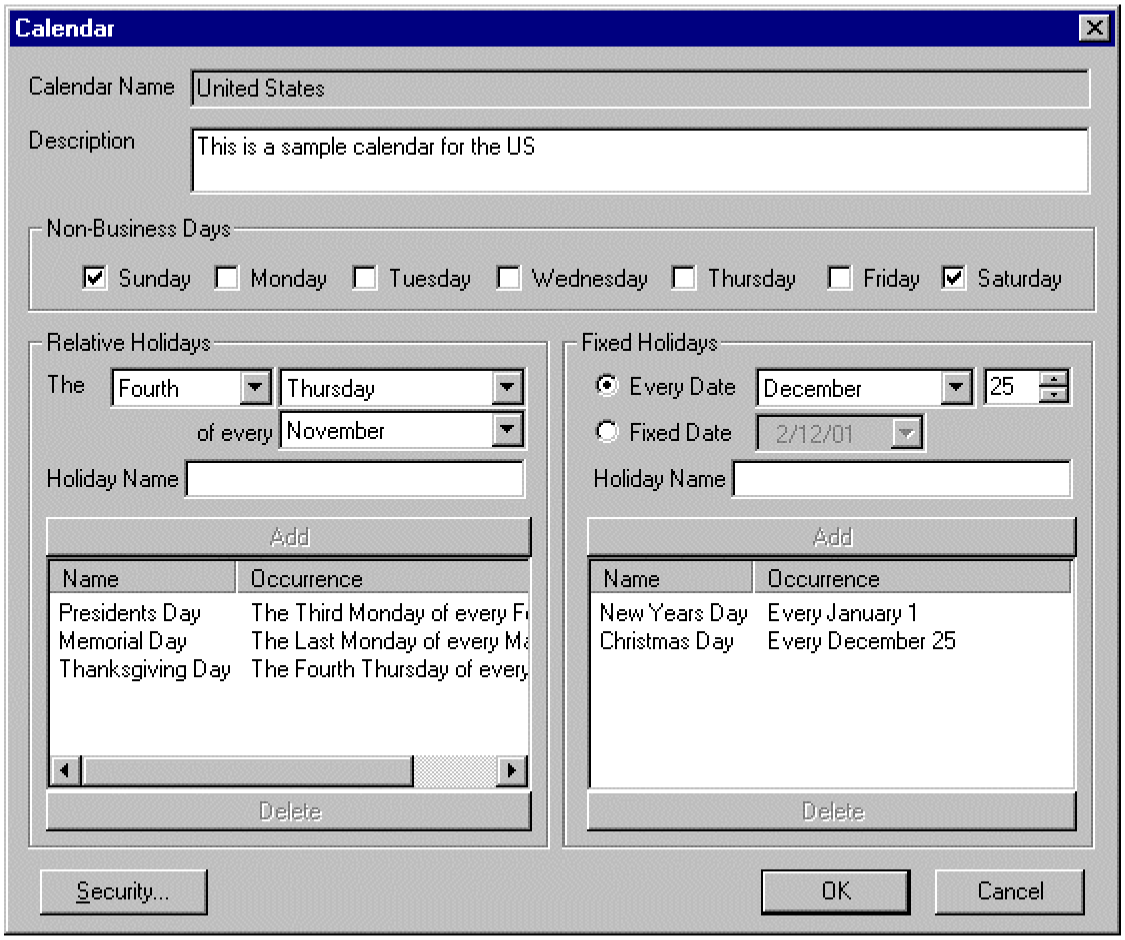
6.2.4 Managing Job Schedules

Job Schedules are date and time oriented triggers that indicate (along with any Business Calendar specified) when a job is to run. Schedules can be created at the Job Scheduler level or you can defer creating the schedule until you also define the job. At this level you can create, edit or delete a Schedule. At the job level, you can also associate or disassociate a schedule from a job. Section 6.5.4 provides complete information on how to manage Job Schedules.

6.2.5 Managing Business Calendars

As the last section mentioned Job Schedules are date and time oriented triggers that indicate when a job may run. By default, ActiveBatch uses the normal 365 day calendar in which any day is a potentially eligible day for the job to run. In other words, ActiveBatch follows the Schedule literally. Sometimes, however, it is important to run jobs based on *Business days.* A Business Day is also known as a "Working Day" (and no one who works for a living needs a definition of that!). An ActiveBatch Calendar contains the specifications for how to determine a working day. ActiveBatch Calendar's are uniquely named and are used by associating the calendar with a job. If a job doesn't have a calendar associated with it then the job isn't provided business day processing.

ActiveBatch allows you to define two (2) types of days to the Job Scheduler by creating a Calendar. They are Holidays, and Non-Working Days (which yields Working Days). A Holiday is defined as a day in which the company is normally open for business but isn't due to the holiday. A Holiday is therefore not considered a holiday if it falls on a non-working day. A Holiday can be defined in a self-relative manner (like the Last Thursday in November is Thanksgiving Day) or via a fixed date without year (like January 1 is always New Year's Day) or with a fixed date and year. The first two specifications are used regardless of year while the last is only used for the applicable year. Non-Working Days are specified by day-of-the-week. So the Saturday and Sunday checkboxes would be selected to indicate non-working days.


Calendar

Calendar Name: United States

Description: This is a sample calendar for the US

Non-Business Days: [✓ Sunday] Monday Tuesday Wednesday Thursday Friday [✓ Saturday]

Relative Holidays

The: Fourth Thursday

of every: November

Holiday Name:

Name	Occurrence
Presidents Day	The Third Monday of every F
Memorial Day	The Last Monday of every M
Thanksgiving Day	The Fourth Thursday of every

Fixed Holidays

[● Every Date] December 25

Fixed Date: 2/12/01

Holiday Name:

Name	Occurrence
New Years Day	Every January 1
Christmas Day	Every December 25

Figure 64 - Defining a Business Calendar

Figure 64 depicts the dialog for creating a business calendar. The sample calendar entered "United States" shows both fixed and relative holiday specifications. While calendars are great for defining holidays you don't have to. You can just define non-business days without also defining any holidays.

Calendar Name: This field uniquely identifies the calendar and is specified when defining a job that is to obey business day rules.

Description: This field provides a brief description of the calendar.

Non-Business Days: This area allows you to indicate which days are non-working days. Click the checkbox next to each day that is to be considered a non-working day. Typically Saturday and Sunday are non-working days.

Relative Holidays: This area allows you to indicate holidays based on a relative specification rather than a fixed date. For example, Thanksgiving Day in the United States is the "Fourth Thursday in November". Three (3) controls allow the relative specification for a holiday. The first control contains the following values: First, Second, Third, Fourth and Last. The second control contains the "day" values: Monday, Tuesday, Wednesday, Thursday, Friday, Saturday and Sunday. The last control contains the months of the year. You may also *Name* the holiday you have entered. Clicking the *Add* button enters the relative specification in the list box below. To delete an already entered holiday, select the date and click the *Delete* button.

Fixed Holidays: This area allows you to indicate holidays that are based on a fixed date. For example, Christmas is December 25. You may specify a fixed date holiday using two methods. If the holiday is to occur on the same date each year, click the *Every Date* radio button and enter the Month and Day of the holiday. If the holiday is to occur on a fixed date for a specific year, click the *Fixed Date* radio button and enter the month, day and year (localized) of the holiday. You may also *Name* the holiday you have entered (limited to 20 characters). When you want to add the fixed holiday, click the *Add* button.

As with Schedules you may specify Security access rights for the named calendar. Typically all ActiveBatch users are expected to be able to access a calendar for Read purposes. To manage security attributes for a Calendar, click the *Security…* button. See Section 6.5.4.7 on "Securing a Schedule/Business Calendar" for more information on the security access permissions that can be assigned to a Calendar.

6.3 Creating Queues

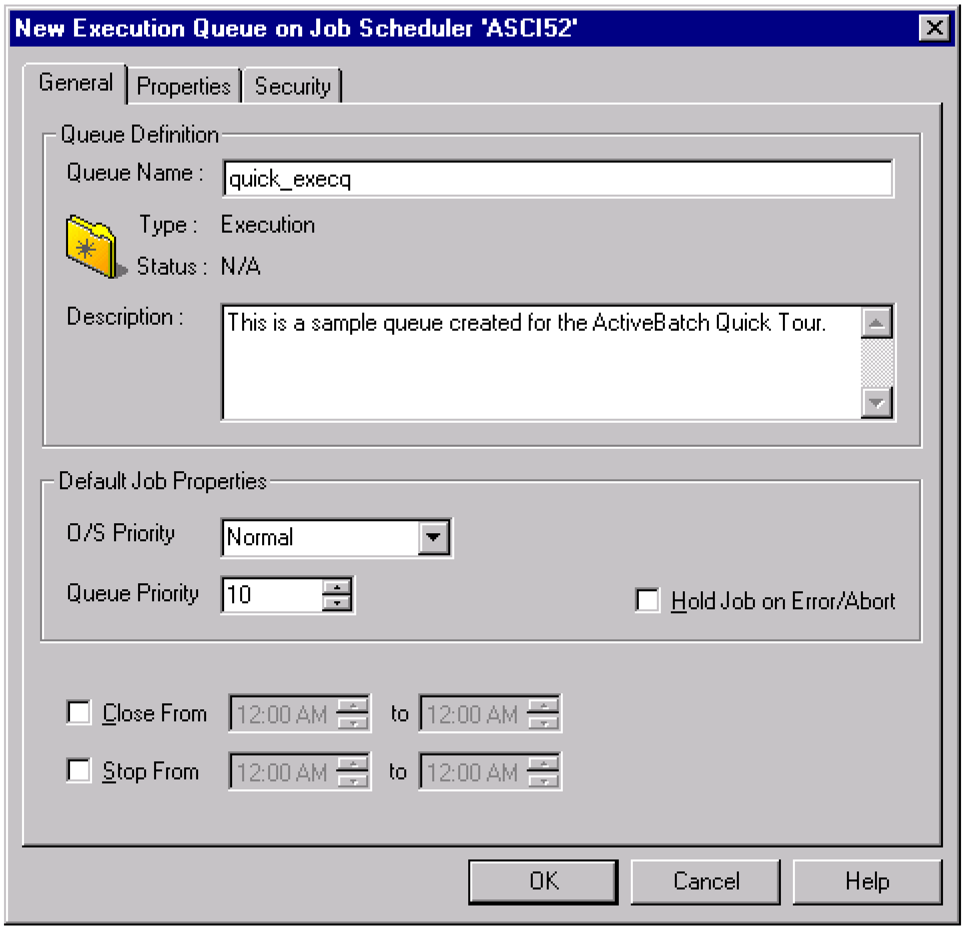
This section will describe how to create Queues and how to exploit Generic Queues.

6.3.1 Creating an Execution Queue

Section 5.3 described the use of the “New Queue (Wizard)” to create an Execution Queue. This section will more fully describe all the options that may be used to create an Execution Queue.

To create an Execution Queue either right click on a Job Scheduler machine icon or use the menu bar “Queue” and select the “New Execution Queue” operation. When you’ve completed the Execution Queue tabs you must click the *OK* button to confirm and apply your selections. Click the *Cancel* button to cancel the New Execution Queue operation. After you’ve confirmed your selection you should receive a message indicating that the New Queue operation succeeded. Depending on the view you’ve enabled, your new queue will instantly appear and should be ready for operations.

6.3.1.1 New Execution/Generic Queue – General


New Execution Queue on Job Scheduler 'ASCI52'

Queue Definition

Queue Name : quick_execq

Type : Execution

Status : N/A

Description : This is a sample queue created for the ActiveBatch Quick Tour.

Default Job Properties

O/S Priority: Normal

Queue Priority: 10

☐ Hold Job on Error/Abort

☐ Close From 12:00 AM to 12:00 AM

☐ Stop From 12:00 AM to 12:00 AM

Figure 65 - New Execution/Generic Queue (General)

The above figure shows the "New Queue" tab sheets. Three "tabs" describe Queues: *General*, *Properties* and *Security*. Both Execution and Generic Queues share the common "General" and "Security" tabs.

Queue Definition: This section of fields describe the common aspects shared by all Queues.

Queue Name: This mandatory field requires the entry of a unique name for this Queue. Queue names are limited to a maximum of thirty-two (32) characters. While you may enter data in upper or lower case all comparisons and searches for this field are performed in a “caseless” mode. You cannot change this property after the Queue has been created.

Type: This read-only field indicates whether the Queue is Execution or Generic.

Status: This read-only field indicates the current status of the Queue.

Description: This optional field allows you to enter data describing the intended use of this Queue.

Default Job Properties: This section of fields provide working defaults (in addition to Queue and Job Policies) for jobs submitted to this Queue.

O/S Priority: This field indicates the default Operating System priority to be assigned to this job. Windows 2000/NT jobs use the keywords: BelowNormal, Normal, AboveNormal, High and Realtime (for NT V4 machines, the “Normals” are considered the same). Other operating systems use a number based system which is also supported by ActiveBatch. A number from 1 to 32 may be specified.

Queue Priority: This field indicates the default priority to be used when a new job is submitted to ActiveBatch. This field should not be confused with the *O/S Priority* field which is used to indicate the priority of the running process. Rather this priority indicates the order a new job is inserted into an ActiveBatch Queue for eventual execution. The higher the value the higher the priority. A range from 1 to 100 is accepted with a typical default of 10.

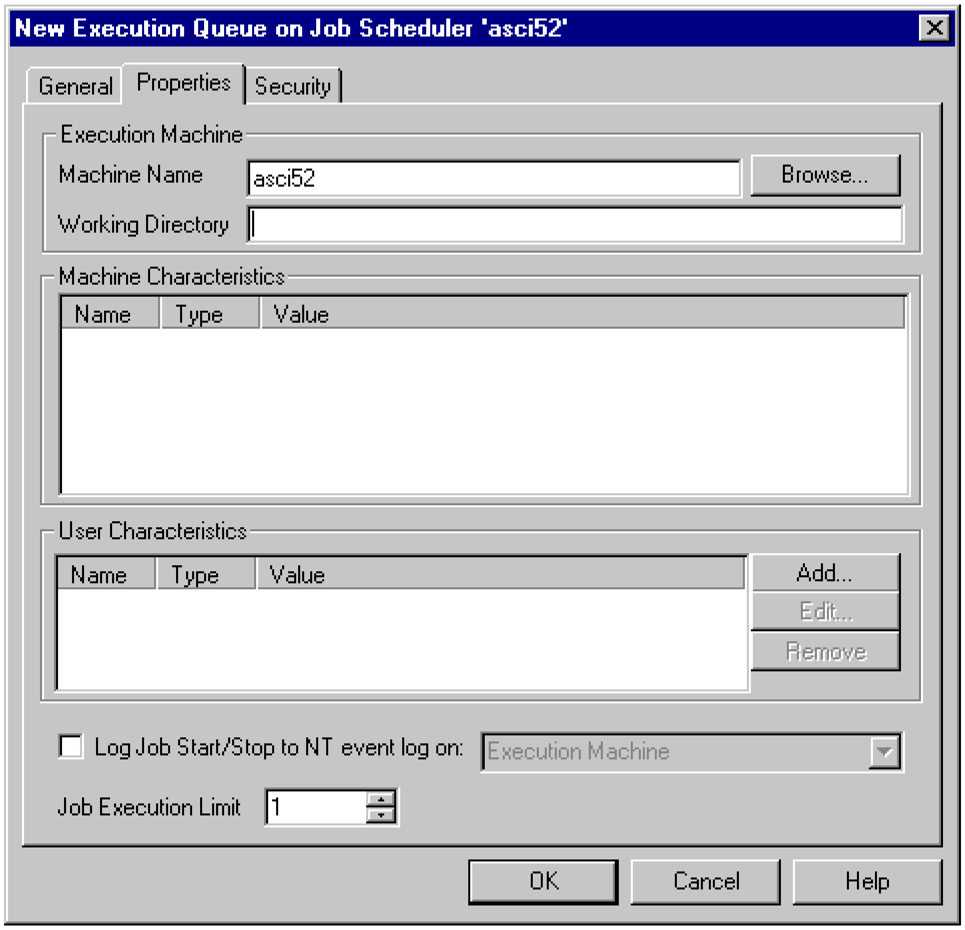
Hold Job on Error/Abort: This checkbox indicates whether a job that completes in error or is aborted should be held on the executing queue. Normally a job is not held.

Operating Times: The next set of fields indicate when jobs may be submitted to the Queue or when jobs may be scheduled for execution.

Close From… to…: This from and to time range indicates when a Queue is closed. Jobs may not be submitted to a closed queue. To enable this feature, click the checkbox and then enter a *from* and *to* time range. From and To times may not be the same.

Stop From… to…: This from and to time range indicates when a Queue is to be stopped. Jobs are not scheduled for possible execution on a stopped Queue. To enable this feature, click the checkbox and then a *from* and *to* time range. From and To times may not be the same.

6.3.1.2 New Execution Queue – Properties


New Execution Queue on Job Scheduler 'asci52'

General | Properties | Security

Execution Machine
Machine Name: asci52
Working Directory:

Machine Characteristics

Name	Type	Value

User Characteristics

Name	Type	Value

☐ Log Job Start/Stop to NT event log on: Execution Machine

Job Execution Limit: 1

Figure 66 - New Execution Queue (Properties)

This tab is used to set specific Execution Queue properties or characteristics.

Execution Machine: This set of fields associates an Execution Queue with an execution machine.

> **Machine Name:** This mandatory field indicates the machine that will be used to execute jobs submitted to this Queue. You may enter any valid machine name or IP address as long as a transport is available on both the Job Scheduler and execution machine to effect

communications. The *Browse* button may be used to select a machine in your Network Neighborhood. The machine entered must be running an ActiveBatch Execution Agent. By default, the client machine name is used.

Working Directory: This optional field may be used to set a default working directory when jobs are submitted to this Queue. The working directory entered must be local and acceptable in syntax to the execution machine. Note: ASCI recommends that you always specify a working directory (either as part of the Queue's properties to obtain a valid default or explicitly as part of this field). The default working directory varies based on the operating system. For Windows 2000/NT, the default is the system temp directory. For UNIX and OpenVMS systems, the default is the user's home directory.

Machine Characteristics: This read-only section indicates the static characteristics of the machine you've selected for this Execution Queue. Generic Queues use this information for selecting the most appropriate execution machine for a job. This section will be blank until the Execution Queue is actually created.

User Characteristics: This section lists the current user characteristics associated with this Queue. Generic Queues use this information for selecting the most appropriate execution machine for a job. To add a user characteristic, click the *Add* button. To edit or remove a user characteristic, select the characteristic from the list and click the *Edit* or *Remove* buttons. See Figure 68 - New Execution Queue (Add User Characteristic) for specific information on this dialog.

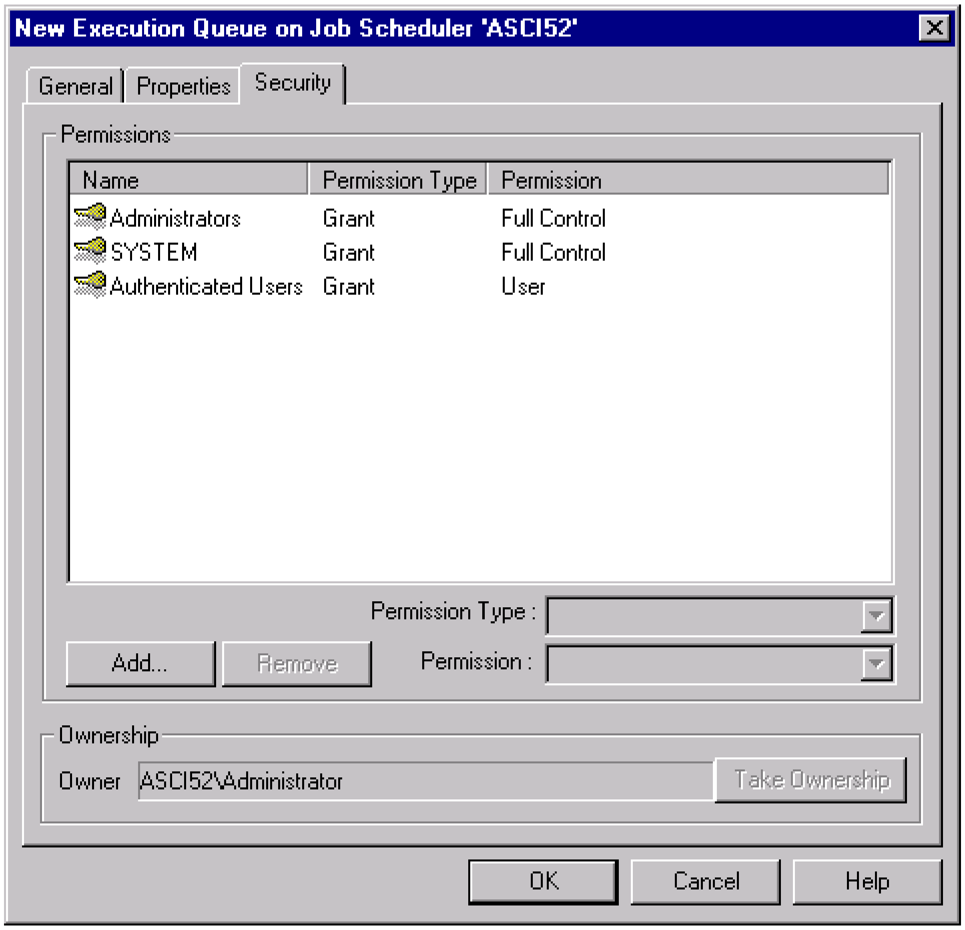
Log Job Start/Stop to NT Event log on: This field indicates whether the Job Scheduler should log all job start and stop events to a machine's NT event log. Clicking the checkbox enables the feature. You may further choose which machine's event log will be used. Valid selections are: Job Scheduler, Client machine, Execution machine, Custom (a valid machine name) or None. Of course, the machine selected must be running Windows 2000/NT. Please note that when selecting an NT event log machine other than the Job Scheduler please read Section 3.3.1 for Job Scheduler Security Service considerations.

Job Execution Limit: This field indicates the maximum number of jobs that may simultaneously execute on this Execution Queue. A number from 1 to 100 may be selected. The typical default is one (1). Please note that too high a number may have an adverse effect on overall execution machine performance.

6.3.1.3 New Execution/Generic Queue – Security

Both Execution and Generic Queues share the same *Security* tab. This tab indicates the security characteristics of the Queue object. The ActiveBatch Administrator may select Queue Policies for an ActiveBatch installation and those policies extend to the security system. Therefore when you create a Queue you may notice that several permissions have already been associated with the object.

To add, edit or remove security access permissions requires the user to have security access to the object.



Name	Permission Type	Permission
Administrators	Grant	Full Control
SYSTEM	Grant	Full Control
Authenticated Users	Grant	User

Permission Type :

Permission :

Owner ASCI52\Administrator

Figure 67 - Queue Security Tab

This tab lists all security access permissions using Windows 2000/NT conventions. To edit an existing account name, select the listed account and change the permission using the **Type of access:** list box. You may select one of the following access permissions:

Access	Description
User	Account is allowed to submit a job to the queue
Manage	Account is allowed to manage the queue and issue operations such as Update, Start/Stop, Open/Close.
Executive	Account is allowed to delete the queue, take ownership and set queue permissions.
Full Control	Account may issue all of the operations mentioned above.
No Access	Account may not access the queue.

Table 2 - Queue Security Access

To add a new account permission, click the *Add* button and follow the dialog as discussed in Section 6.3.1.5. To remove an existing account name, select the listed account and click the *Remove* button.

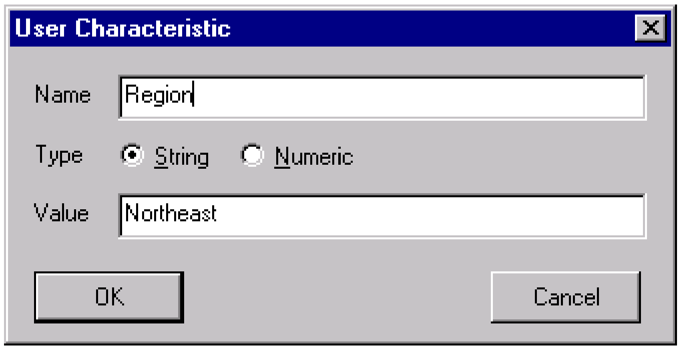
Full Control is simply a convenient way of granting all access permissions at one time. Any combination of access permissions are allowed (except No Access). By default, ActiveBatch provides the following access permissions when a queue is created.

Administrators	Executive, Manage
Built-in System	Full
Authenticated Users	User

This policy can be changed via the ActiveBatch MMC snap-in.

The owner of a queue is the user who first creates the Queue. The Queue owner is always implicitly granted “Full Control” access. To change ownership, click the *Take Ownership* button and the change is made (remember that in Windows 2000/NT you cannot specify a user to become owner rather a user “takes” ownership.

6.3.1.4 Add User Characteristic (Queue)


User Characteristic

Name: Region

Type: (●) String ( ) Numeric

Value: Northeast

Figure 68 - New Execution Queue (Add User Characteristic)

The above figure is displayed when you want to add or edit a user characteristic. User characteristics are uniquely named within a single queue. Typically you will add the same named user characteristic to one or more Execution Queues associated with a Generic Queue. The purpose of user characteristics is to allow a batch job that has been submitted to a Generic Queue to directly influence which machines should be considered eligible for running a job.

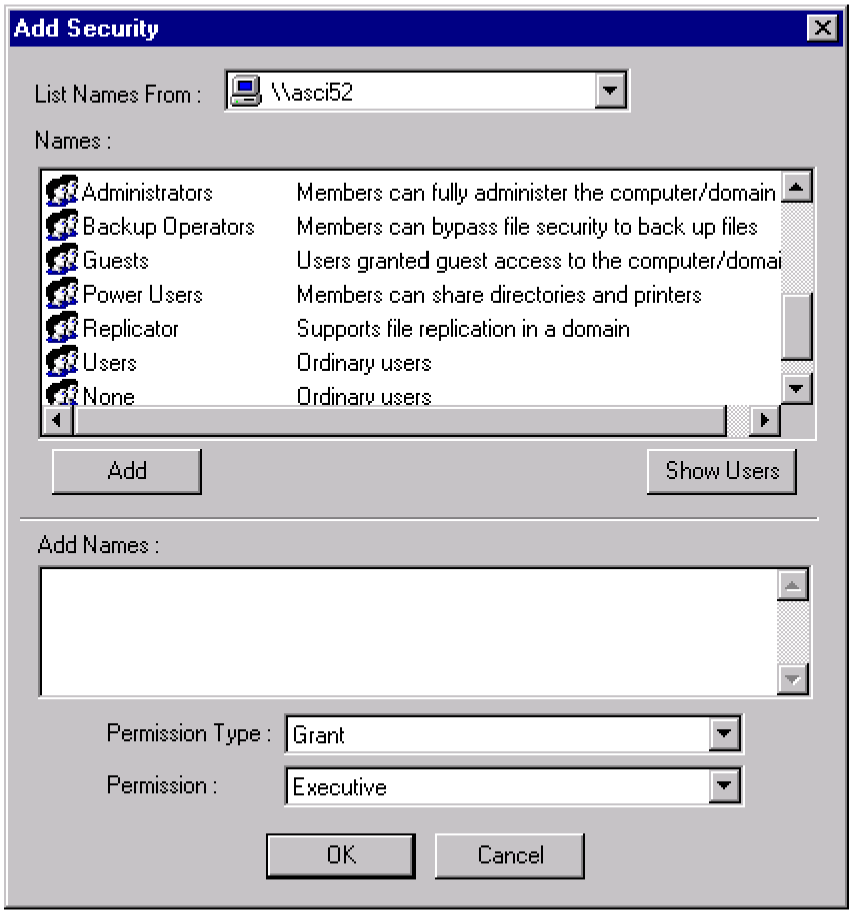
Special reserved user characteristics beginning with the prefix ABAT$ are reserved to ASCI. ABAT$QAFFINITY allows a user to select a preferred execution queue when that queue is a member of a generic queue. This functionality is most appropriate when the generic queue represents a cluster.

Name: Enter a name of your choosing which will represent this characteristic. The characteristic must be uniquely named only within the same queue.

Type: You must select whether the characteristic value is a numeric or string value.

Value: You must enter a value that will be associated with this characteristic.

6.3.1.5 Add Security Dialog (Queue)



List Names From : \\asci52

Administrators	Members can fully administer the computer/domain
Backup Operators	Members can bypass file security to back up files
Guests	Users granted guest access to the computer/domai
Power Users	Members can share directories and printers
Replicator	Supports file replication in a domain
Users	Ordinary users
None	Ordinary users

Permission Type : Grant

Permission : Executive

Figure 69 - Add Security Dialog (Queue)

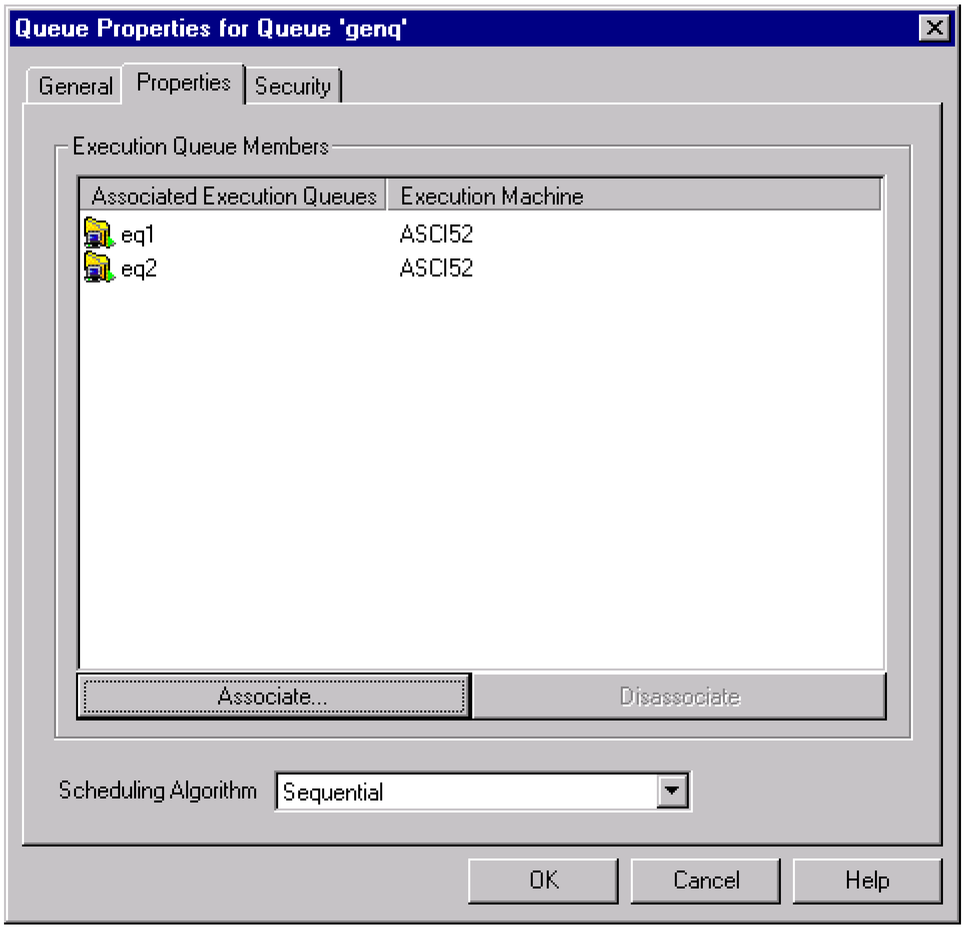
The figure above displays the Security dialog for a Queue. The dialog is similar to that of other Windows 2000/NT objects. You may only select accounts from the Job Scheduler machine or domain(s). The accounts are then listed in the **Names:** list box. To fully display all user names you may click on the *Show Users* button. To select the permission to be associated with an account, you click on the **Permission Type** list (you may Grant or Deny access) and then the **Permission** list (for the specific access permission). When you press the *Add* button the selected account is merged with the displayed permissions and added to the **Add Names:** area. You may add multiple accounts in this fashion. Click the *OK* button to confirm and apply your selections.

Click the *Cancel* button to cancel security changes. When you confirm and apply your selections, your account/permissions will be added and/or merged into the security permissions of the Queue object as per Windows 2000/NT guidelines. See Section 6.3.1.3 for more information on Queue Security and access permissions.

6.3.2 Creating a Generic Queue

The concept of Execution Queues is pretty straightforward. After all, you know you want to run jobs on machines so selecting those machines isn't exactly radical. On the other hand, just what is the usefulness of a Generic Queue? Well before we learn how to create one let's understand why we might want one. A Generic Queue is composed of one or more Execution Queues. An Execution Queue can be a member of one or more Generic Queues. The membership of an Execution Queue to a specific Generic Queue can be as static or dynamic as you wish. This abstraction layer between the Job and the machine that it actually runs on can be very useful. For example, you have a job that could run perfectly well on one of three machines. If you submitted the job directly to an Execution Queue – that's it. The job will only execute on that machine. Period. If the machine isn't available, the job will wait. If the machine is incredibly overloaded, too bad. Of course, you could move the job to another Queue but that isn't why you bought ActiveBatch. A Generic Queue is a virtual machine. The actual machine that will be selected can be deferred to the last moment. This means the job can execute on the most appropriate machine available. You can also specify selection criteria as well as match machine/user characteristics of the job to the machine/user characteristics of the Execution Queue. So if you need to run a job on a Pentium II of at least 500 Mhz you can. Let's examine the Generic Queue tabs and learn more about Generic Queues.

As we learned with Execution Queues, three (3) tabs are provided to define a Queue. The *General* and *Security* tabs are common to all Queues. So let's concentrate on the Generic Queue *Properties* tab.



Execution Queue Members

Associated Execution Queues	Execution Machine
eq1	ASCI52
eq2	ASCI52

Scheduling Algorithm: Sequential

Figure 70 - New Generic Queue (Property)

The above figure consists of a list and a field. The list displays all Execution Queue(s) which are members of this Generic Queue. For a new Generic Queue, this list will be empty. To add members to this Generic Queue, click the *Associate* button. A new dialog box is displayed listing all available Execution Queues for you to select. Simply select one or more Queues and click the *OK* button. Those queues will be added to the list. To remove existing members from this Generic Queue, simply select the member(s) and click the *Disassociate* button.

Scheduling Algorithm: This field indicates what algorithm should be used to dispatch jobs to eligible Execution Queues. An eligible queue is one that meets all requirements for the job and is available for execution at the moment the job is actually dispatched to an Execution Queue. Possible choices are: *Sequential*, *Round Robin* and *Workload Balancing*. **Sequential** means that the first Execution Queue is used for all job execution dispatches until the Queue has reached its maximum job limit and then the next queue is selected. *Round Robin* causes jobs to be dispatched to each Queue in turn. *Workload Balancing* causes ActiveBatch to analyze the run-time aspects of the execution machine and uses the best (least used) execution machine available. CPU and Memory are among the criteria used in workload balancing.

6.4 Managing Queues

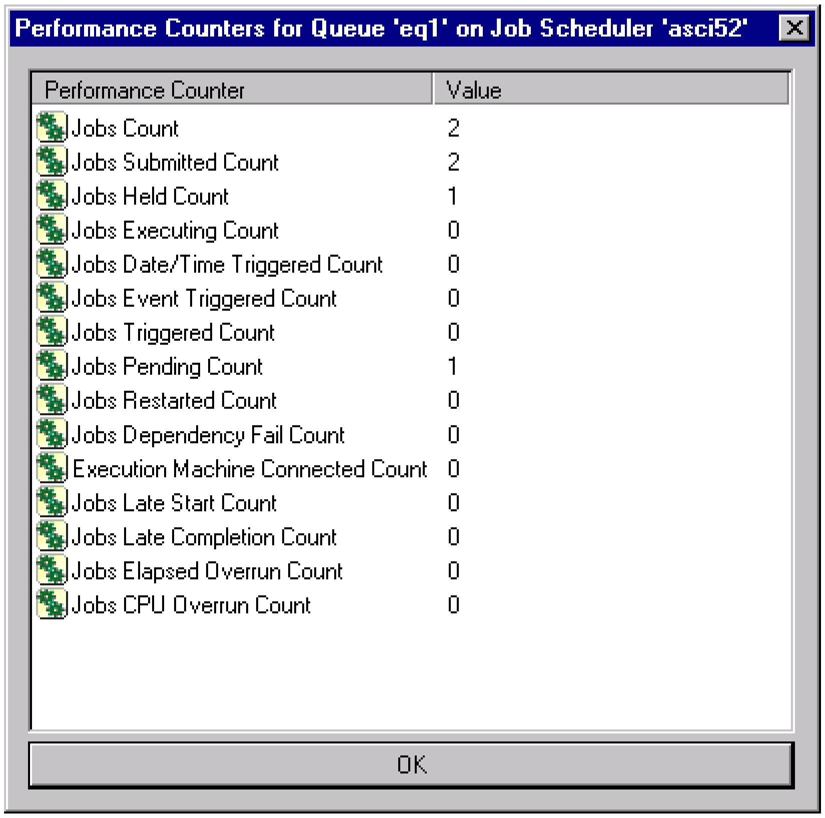
Once you've created a Queue you're halfway home to executing a Job. However, while we're on the subject of Queues let's complete our discussion by illustrating some types of operations we can perform on Queues.

6.4.1 Modifying a Queue

Modifying an existing Queue is pretty simple. Just right-click on the Queue you want to modify and select *Properties*. The three (3) tabbed dialog box will appear allowing you to change most of the Queue's properties. One property you cannot change is the Queue's name and type. Modifying a Queue requires that you hold "Manage" access permission to the Queue object.

6.4.2 Queue Counters

ActiveBatch maintains performance counters for all Queue objects.


Performance Counters for Queue 'eq1' on Job Scheduler 'asci52'
OK

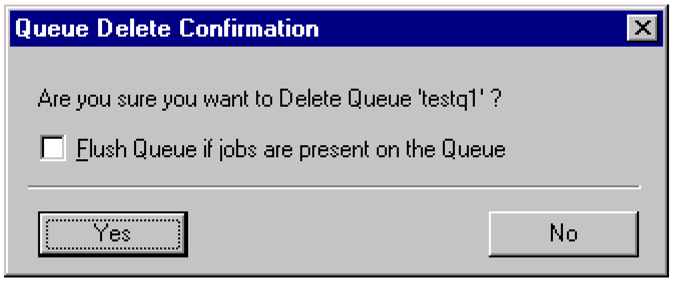
Performance Counter	Value
Jobs Count	2
Jobs Submitted Count	2
Jobs Held Count	1
Jobs Executing Count	0
Jobs Date/Time Triggered Count	0
Jobs Event Triggered Count	0
Jobs Triggered Count	0
Jobs Pending Count	1
Jobs Restarted Count	0
Jobs Dependency Fail Count	0
Execution Machine Connected Count	0
Jobs Late Start Count	0
Jobs Late Completion Count	0
Jobs Elapsed Overrun Count	0
Jobs CPU Overrun Count	0

Figure 71 - Queue Counters

The counters fall into two categories: Operational and Historical counts. Operational counts are those which accurately describe the current state of the queue. Historical counts are those which are accumulated while the product has been actively running and should be considered volatile between product startups.

6.4.3 Deleting a Queue

To delete an existing Queue, either right-click to select the Queue icon and choose *Delete…* or select the Queue icon and choose *Queue/Delete…* from the menu bar. Deletion of a Queue requires that no jobs of any kind be on queue. ActiveBatch will detect whether there are any jobs on queue and allow you to flush them as part of the deletion process.


Queue Delete Confirmation

Are you sure you want to Delete Queue 'testq1' ?

☐ Flush Queue if jobs are present on the Queue

Figure 72 - Delete Queue Confirmation

The figure above is presented in all cases to ensure you really want to delete the selected queue. If jobs are present, you should also click on the checkbox to flush those jobs from the Queue. Please note that when you delete a Queue, that Queue will also be removed as a member from all Generic Queues it may be associated with. You cannot delete the OnDemand Queue. Deletion of a Queue requires that you hold "Executive" access permission to the Queue object. Read Section 6.4.6 on Flushing a Queue.

6.4.4 Starting and Stopping a Queue

Stopping a Queue prevents any jobs that are currently on-queue and in a pending state from additional processing. For an Execution Queue this means those jobs will not be allowed to start execution. If a job is already executing, stopping a Queue does not affect the running job. For a Generic Queue, pending jobs are not dispatched to member Execution Queue(s). Stopping a Queue is therefore beneficial when you know a machine will be subjected to some downtime or a reboot. Another advantage to stopping a member Execution Queue is that jobs submitted to a Generic Queue won't be dispatched to the stopped Queue.

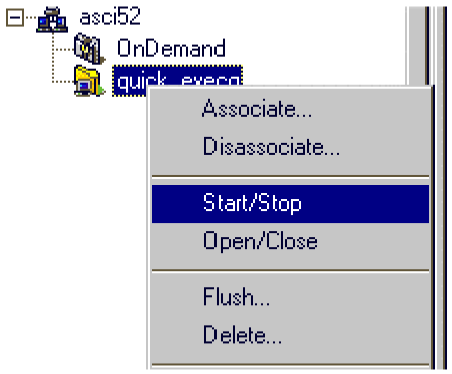
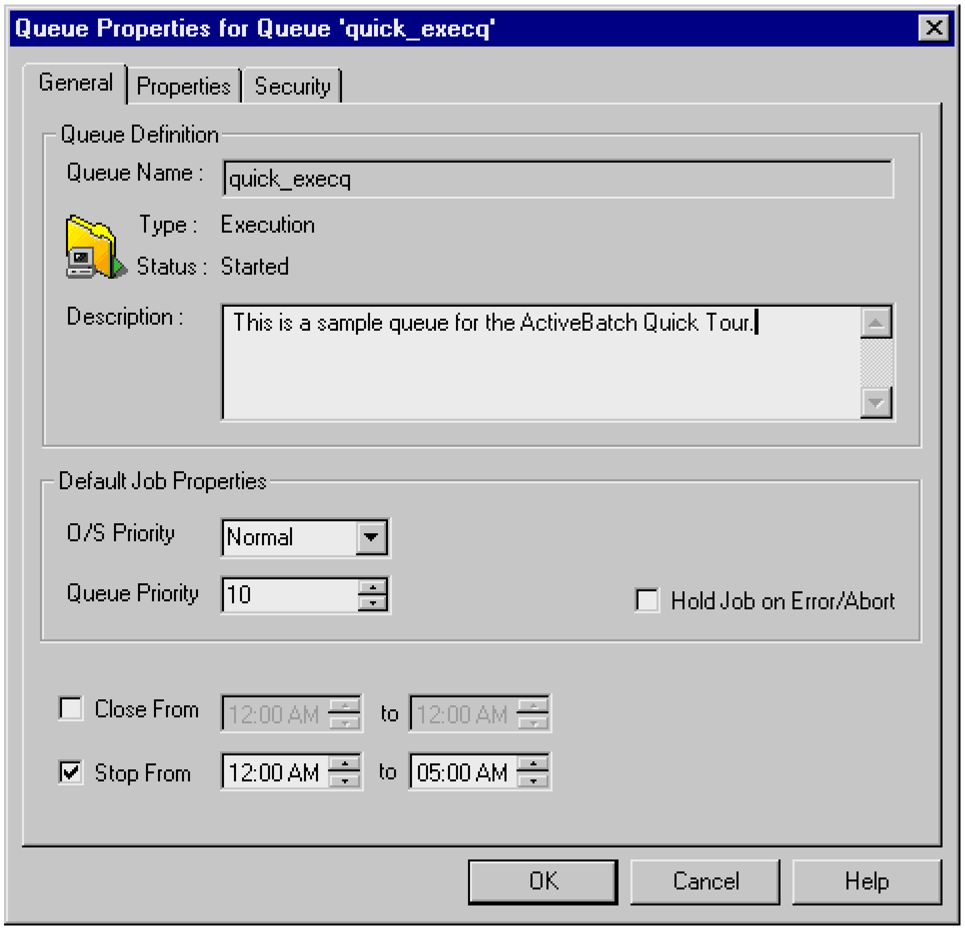


Figure 73 – Stopping/Starting a Queue

To stop a started Queue, either right-click on the selected Queue icon and choose Start/*Stop* or select the Queue icon and choose *Queue/Start/Stop* from the menu bar.

Starting a Queue is of course the same general actions except that the *Start* menu operation is chosen. Normally when you start an Execution Queue the execution machine will be up and available. If it isn't the Queue will enter a "Starting" state and remain in that state until the execution machine becomes available. Jobs are not dispatched to a Queue in a "Starting" state. The *Start/Stop* operation is an in-context operation. That means ActiveBatch will select the proper operation depending on the current state of the Queue (e.g. if it's started, you can stop the Queue).


Queue Properties for Queue 'quick_execq'

Queue Definition

Queue Name : quick_execq

Type : Execution

Status : Started

Description : This is a sample queue for the ActiveBatch Quick Tour.

Default Job Properties

O/S Priority: Normal

Queue Priority: 10

☐ Hold Job on Error/Abort

☐ Close From 12:00 AM to 12:00 AM

☑ Stop From 12:00 AM to 05:00 AM

Figure 74 - Stopping a Queue (Timed)

Remember that you can also automatically stop (and start) a Queue based on the Queue's properties. The figure above is taken from the Queue General tab. This is especially useful when you know a machine won't be available during certain hours.

To start or stop a Queue, you must hold "Manage" access permission on the Queue object.

6.4.5 Open and Close a Queue

Where as stopping and starting a Queue deal with whether jobs are dispatched or executed, opening and closing a Queue deals with whether jobs can be submitted to a Queue by a user. Typically Queues are open to job submissions regardless of whether the Queue can actually support job execution. A “Closed” Queue indicates that jobs will not be accepted for submission. This feature isn’t used very often since most users want to allow job submissions. However to close or open a Queue, you can either right-click on the Queue icon and select *Open/Close* or select the Queue icon and choose the *Queue/Open/Close* operation from the menu bar. The *Open/Close* operation is an in-context operation. That means ActiveBatch will select the proper operation depending on the current state of the Queue (i.e. if it’s open, you can close the Queue).

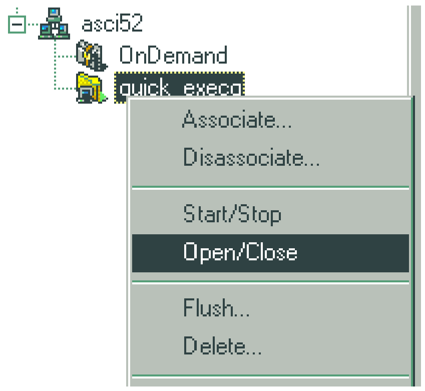
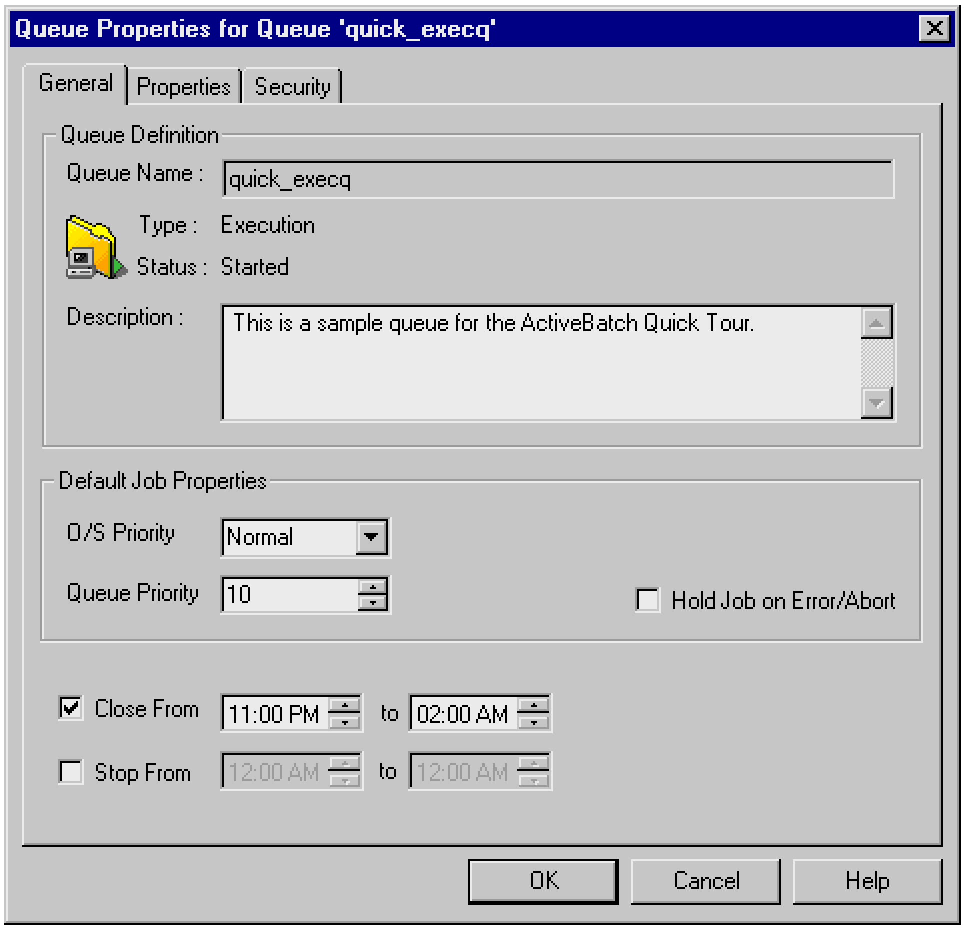


Figure 75 - Opening or Closing a Queue

Remember that you can also automatically schedule open and close operations on a per queue basis as part of the Queue’s properties. Note the *Close From* field below in Figure 76.



Queue Definition

Queue Name : quick_execq

Type : Execution

Status : Started

Description : This is a sample queue for the ActiveBatch Quick Tour.

Default Job Properties

O/S Priority: Normal

Queue Priority: 10

☐ Hold Job on Error/Abort

☑ Close From 11:00 PM to 02:00 AM

☐ Stop From 12:00 AM to 12:00 AM

Figure 76 - Restrict Job Submission (Timed)

To open or close a Queue, you must hold "Manage" access permission on the Queue object.

6.4.6 Flush a Queue

This operation is used to quickly delete all jobs that have been submitted or dispatched to a specific Queue. Rather than delete each job individually you can simply flush the Queue. When you flush a queue you lose all job history concerning the flushed jobs.

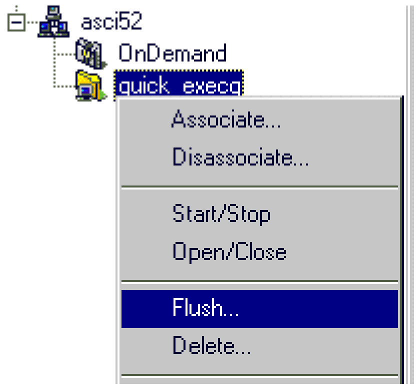
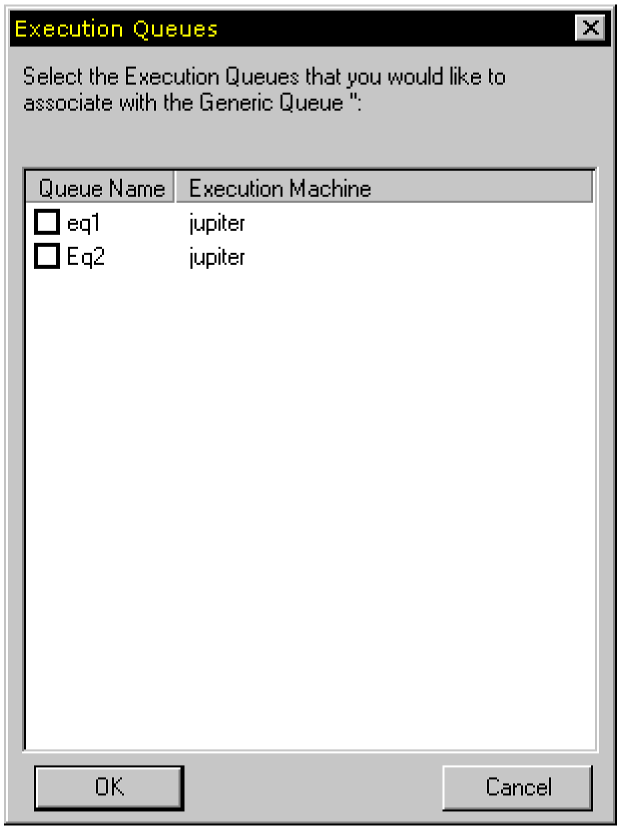


Figure 77 - Flush a Queue

To flush a Queue, you must hold “Manage” access permission on the Queue object.

6.4.7 Associating/Disassociating a Member Execution Queue

This operation allows you to associate/disassociate one or more Execution Queue(s) as members to/from a Generic Queue. To associate Execution Queue(s) to a Generic Queue you must either: right-click on the Generic Queue and then select *Associate…* or select the Generic Queue and choose *Queue/Associate…* from the menu bar.



Select the Execution Queues that you would like to associate with the Generic Queue '':

Queue Name	Execution Machine
☐ eq1	jupiter
☐ Eq2	jupiter

Figure 78 – Associating/Disassociating an Execution Queue Dialog

The main list shows all available Execution Queues. Next to each Queue is a checkbox. If the box is checked, the Queue is already a member of this Generic Queue. To add members, simply click the checkbox next to each desired Queue. When you're done, click *OK* to confirm and apply your selections.

To disassociate existing members, the operations are very similar. Choose *Disassociate* from the menu selection. Figure 78 will appear however this time, only member Queues are listed with a clicked checkbox next to each one. To perform the disassociation, *unclick* the checkbox next to each Queue that you no longer want to be a member of this Generic Queue. Click *OK* to confirm and apply your selections.

To associate/disassociate member Queues, you must hold “Manage” access permissions on the Generic Queue object.

6.5 Job Definition

If Queues are the foundation of ActiveBatch, Jobs are the primary beneficiaries. After all, the entire ActiveBatch system is designed to efficiently execute jobs. A Job is a script or program that will be executed on a compatible machine, with or without operator intervention, in a reliable and secure manner. Typically a Job is thought to run in an unattended mode in that the desktop itself is not occupied with the task of running the job. In fact, with ActiveBatch, the desktop need not be logged on. Jobs in ActiveBatch can be as simple or as complex as needed to perform the tasks you need to successfully complete.

The next section will discuss the creation of a Job and take you through all the Job tab pages. Succeeding sections will provide in-context examples as to what features you can invoke to solve various situations you may be confronted with.

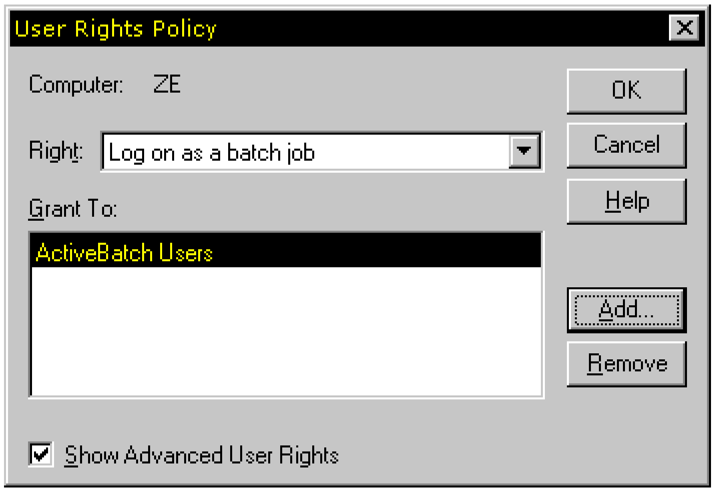
6.5.1 Batch Job Security Requirements

Note: New ActiveBatch Administrators should read this section in its entirety.

Submitting a Job to ActiveBatch requires at least one Execution Queue. The creation of Execution Queues is covered in Section 6.3.1 (In the Quick Tour it's Section 5.3). Before we delve into the actual mechanics of submitting a Job let's discuss some requirements. First, all Jobs in ActiveBatch run under a security context. Typically that context will be the submitting user's username and password. You can specify a different username and password for execution of the Job, however, all credentials supplied are authenticated by the execution machine. As ActiveBatch supports different operating systems and platforms, different authentication is used, however, a username and password is common to all systems. Next, for Windows 2000/NT systems, an additional requirement must be met. *The user who desires to execute a batch job on a machine must be granted the "Logon as a Batch Job" right. This right must be locally granted to the machine the job will execute on.* Most users are participants of some type of domain and so their accounts are typically domain accounts. Please note that the local aspects of the "Logon as a Batch Job" right are important. A typical mistake is to grant this right at the domain level. Once again the right must be granted at the local machine level, specifically on the execution machine. Non-Windows 2000/NT machines don't need to worry about this right.

The best approach is to allow the Execution Agent to automatically grant the "Logon as a Batch Job" right to users who you have already been authorized as having the ability to execute batch jobs. If you enabled this facility as part of the ActiveBatch installation then you don't need to worry about granting this right manually. The Execution Agent registry value "ExecAutoRight" controls this facility (see Section 3.2.1 for more information).

If you expect to grant more than one person access to executing batch jobs, you may want to consider creating an ActiveBatch local group, assigning the “Logon as a Batch Job” right to the group and then adding user members to the group. Figure 79 uses the “Windows User Manager” and illustrates how this would be done. Note that you must click the “Show Advanced User Rights” checkbox to obtain and set the “Log on as a batch job” right. In the example below the group *ActiveBatch Users* has been created and is now assigned that right on the local computer named ZE. You could now simply add domain members/groups to this local group to complete the required right assignment.


User Rights Policy

Computer: ZE

Right: Log on as a batch job

Grant To:

ActiveBatch Users

☑ Show Advanced User Rights

Figure 79 - Adding the BatchLogon Right

6.5.2 Batch Job Files and Run-Time Considerations

ActiveBatch allows you to run batch jobs across a wide range of heterogeneous platforms.

First, lets consider what files may be submitted as batch job scripts. ActiveBatch supports running program images (.EXE) and, on Windows 2000/NT machines, any file may be specified as long as the extension has a known file association. For example, test.bat, test.vbs, test.xcp, etc. If a script file doesn't have a known file association you can still submit the file by specifying the program that would run the file as the main input job file and the script as its input parameter. For OpenVMS machines you may run a program image (.EXE) or a Command Procedure (.COM). For UNIX machines, you may run a program image (.EXE) or a script file.

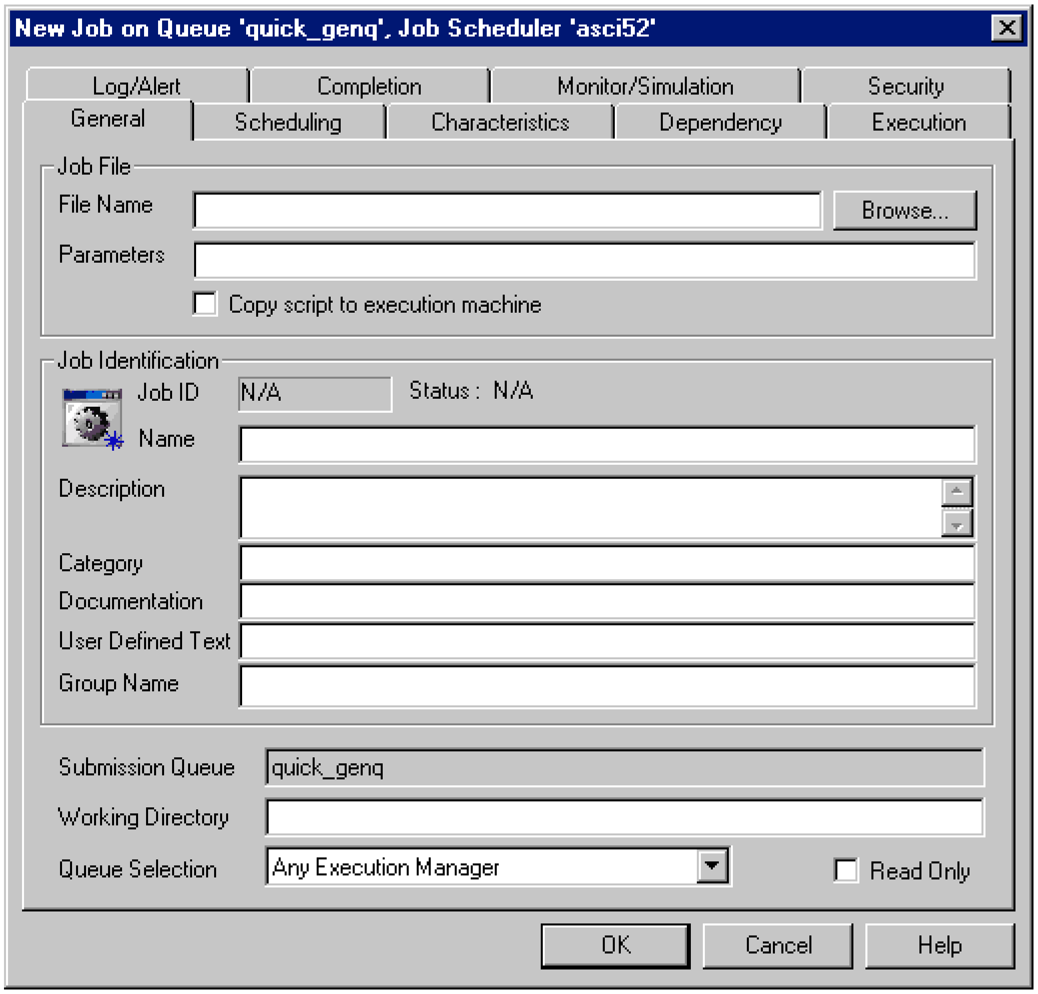
Next, depending on the Operating System some operational differences may exist between the Interactive and Batch environments. This is particularly true for batch jobs submitted on Windows 2000/NT systems. When you log into a Windows system, the interactive logon process re-establishes any persistent network connections, runs any applications found in the Startup folder and executes a logon script if one exists. For a batch job, Windows does not perform those operations. Some aspects of Windows are inherently shared. For example, if you are logged on at the time you run your batch job your network drive letters will be visible. This may be disconcerting when you then expect those same driver letters to continue to appear even when you are no longer interactively logged in. The best practice is for your batch job to establish any operational environment needs as part of the batch job.

Operating System	Feature	Description
Windows 2000/NT	User Profile	HKEY_CURRENT_USER is not available to batch jobs through this hive key on Windows NT.
	Network Connections	Network drive letters are not established by Windows for batch jobs. If your batch job needs access to a network share you should either specify a UNC path or create the network drive as part of the Batch job.
	Login Script	Not implicitly executed for Batch jobs.
OpenVMS	F$MODE	Use of this lexical will produce an "Interactive" string value. To programmatically determine that a command procedure is executing in ActiveBatch batch mode, check for the presence of the logical ABAT_JOBID.

Table 3 - Operating System Batch Job Considerations

6.5.3 Submitting a Job

To define or submit a Job, either right-click on the Queue icon you would like to submit the job to and select *New Job* or select the Queue icon and choose *Job/New Job* from the menu bar.


New Job on Queue 'quick_genq', Job Scheduler 'asci52'

Log/Alert | Completion | Monitor/Simulation | Security
General | Scheduling | Characteristics | Dependency | Execution

Job File
File Name:
Parameters:
☐ Copy script to execution machine

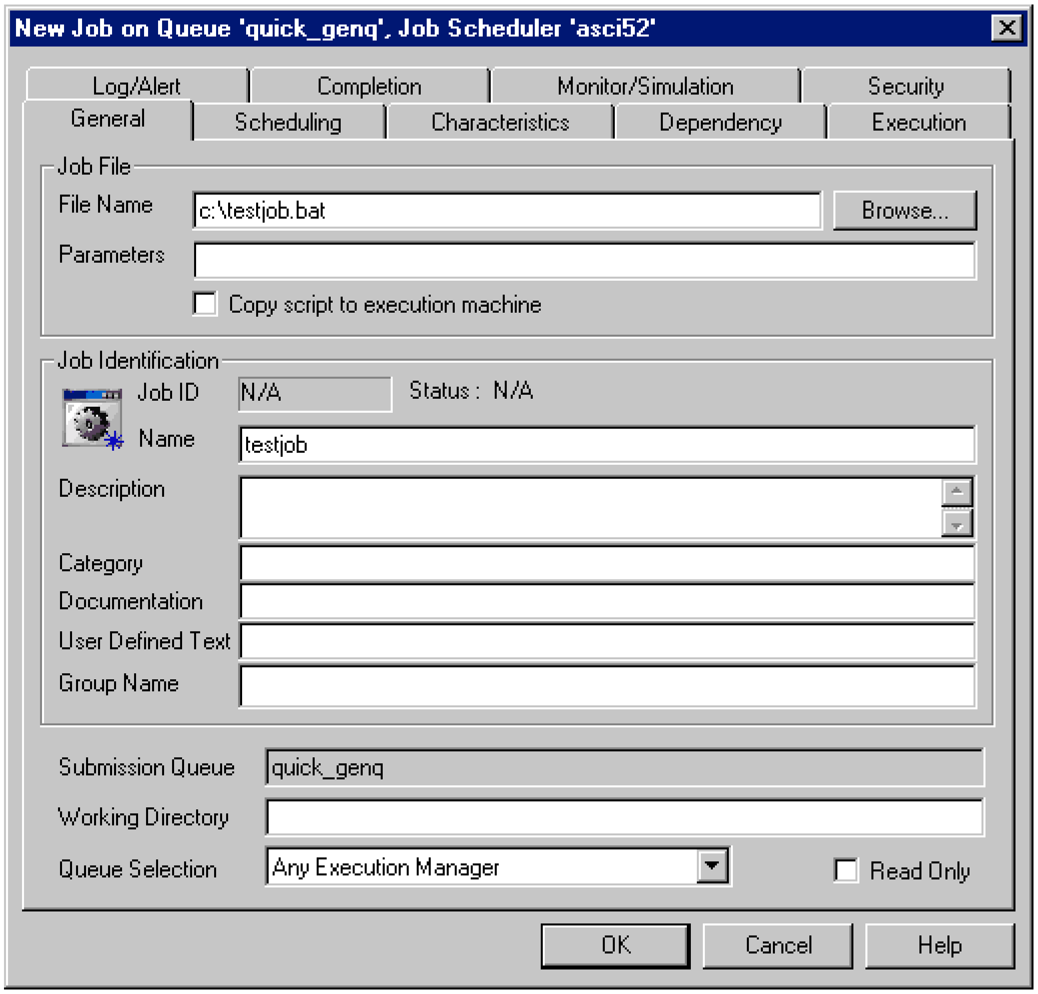
Job Identification
Job ID: N/A   Status : N/A
Name:
Description:
Category:
Documentation:
User Defined Text:
Group Name:

Submission Queue: quick_genq
Working Directory:
Queue Selection: Any Execution Manager   ☐ Read Only

Figure 80 - New Job Property Tabs

The figure above is displayed containing the following tabs: General, Scheduling, Characteristics, Dependency, Execution, Log/Alert, Completion, Monitor/Simulation and Security. The next sections will detail each tab. When you have completed all required information, click *OK* to confirm and submit the job to the queue you selected. **Note: All job changes are only applied when the OK button on the Job Property sheet is clicked.**

6.5.3.1 New Job – General



Job File

File Name: c:\testjob.bat

Parameters:

☐ Copy script to execution machine

Job Identification

Job ID: N/A   Status : N/A

Name: testjob

Description:

Category:

Documentation:

User Defined Text:

Group Name:

Submission Queue: quick_genq

Working Directory:

Queue Selection: Any Execution Manager   ☐ Read Only

Figure 81 - New Job – General

Figure 81 begins the definition and submission of a new job. The dialog box is composed of several areas.

Job File: This area requests the Job's file specification and any parameters which are to be input to the Job on execution.

> **File Name:** This mandatory field represents the main job file (or step) that is to be executed. Unless the *Copy script…* checkbox is enabled, the file specification must be accessible from the eventual execution machine using this specification. A typical mistake is to specify a local drive and directory on a client machine when a different execution machine will be used to actually run the job. You can specify a UNC path for the job file specification. For frequent or recurring jobs the Job's file should reside on the execution machine for best performance. Your security context will be used to access the file so please make certain the file is accessible by you.
>
> **Parameters:** This optional field can be used to pass input parameters to the Job. All data specified is passed as a string with any embedded spaces entered.
>
> **Copy script to execution machine:** This checkbox controls whether the file specification is considered local or accessible from the client and should be copied to the Job Scheduler machine and then ultimately to the Execution machine. The file size must be less than the System policy *CopyScriptMaxSize*.

Job Identification: This area is used to identify the Job.

> **Job ID:** This read-only field contains the Job's unique ID. This field is filled in after the Job has been successfully submitted into the system.
>
> **Status:** This read-only field contains state information about the Job. Possible states are:

State	Description
Pending	The job is awaiting execution. Possible sub-states are: **QueueBusy** – the Queue is not available for Job execution at this time. **WaitingDependency** – waiting on one or more dependencies to complete **WaitingMultipleInstances** – waiting for the active job instance to complete (job tolerates only one active instance) **WaitingTrigger** – waiting for a Trigger operation **Waiting for Restart** – a job with Restart on Reboot is waiting for the Execution Machine to

	become available **Waiting for JobActive** – a template job with *If Job Active Wait* is waiting for the currently active instance to complete **Deferred** – waiting for scheduled execution date/time **Marked for Abort/Delete –** the template job has been marked for abort/delete and will be aborted/deleted when it's instances has completed execution **Not Queued** – this is a transient state that you may see while the job is being submitted to a queue.
Held	The job is held waiting for manual release.
Executing	The job is currently executing. Possible sub-states are: **Executing** – main job step is executing **ExecutingPre** – pre job step is executing **ExecutingPost** – post job step is executing **Paused** – job is paused on execution machine
Completed	The job has completed processing. Possible sub-states are: **Success** – job completed successfully **Failure** – job completed in failure **Aborted** – job was aborted

Table 4 - Job States and Sub-States

Name: This mandatory field represents the name of the Job. You should enter a unique job name to avoid confusion. By default, ABATAdmin uses the filename portion of the Job's File Name.

Description: This optional field can contain free-form text describing the nature and conduct of this job.

Category: This optional field is used to categorize the job.

Documentation: This optional field is used to denote a reference to the job in an operator's runbook or other documentation concerning the running of the job.

User Defined: This optional field can be set by the Job's Author.

Group Name: This optional field can be used to name a related group of Job's. The advantage to naming this group is that operations can be performed on a group of job's.

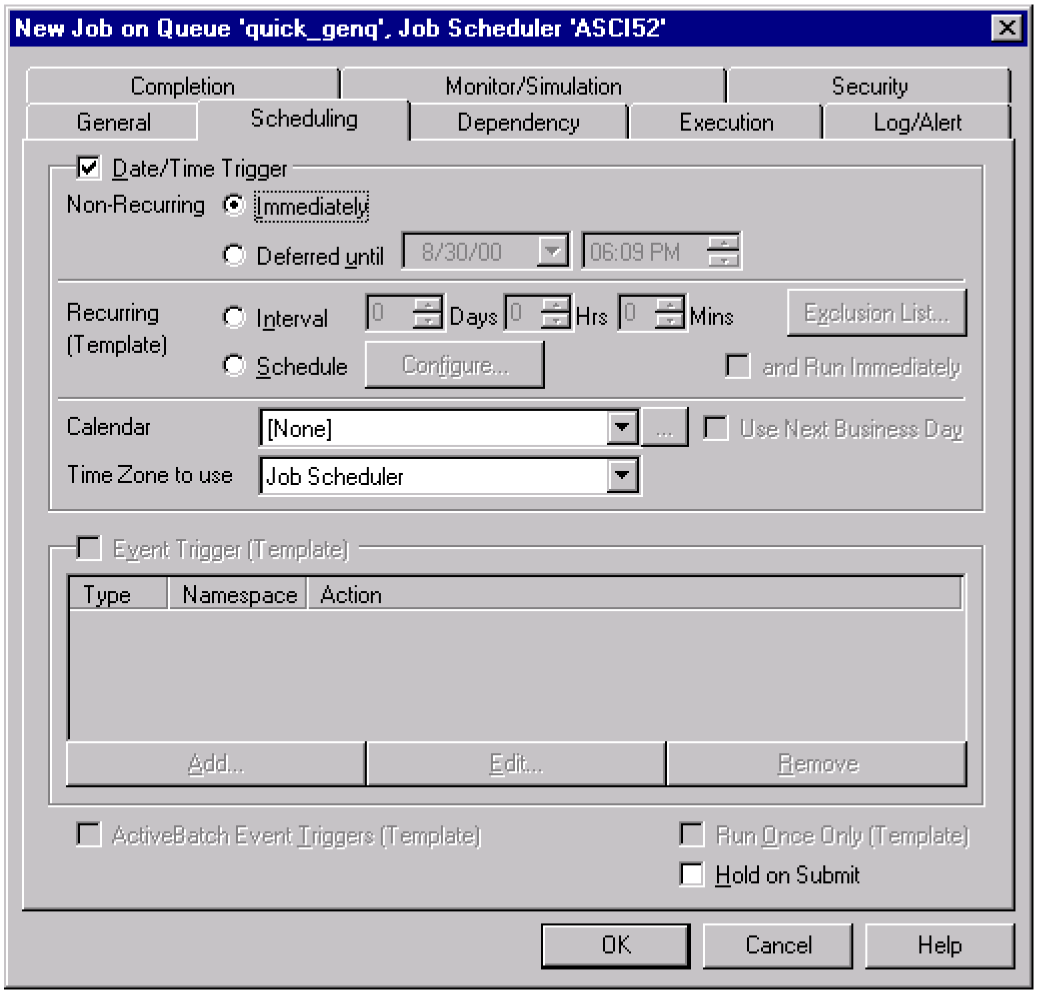
Submission Queue: This read-only field contains the Queue name that this Job is being submitted to.

Working Directory: This optional field can contain the initial directory associated with the Job when it is executed. The directory specification must be accessible from the execution machine. By default, the selected Queue's Working Directory is used.

Queue Selection: If the *Submission Queue* is a Generic Queue then this field indicates an initial job-to-execution queue selection criteria. Possible choices are: Any and All. "Any" means that any eligible Execution Queue members may be used to run this Job in accordance with the Generic Queue's *Scheduling Algorithm*. "All" means that all eligible Execution Queue(s) members should be used to run this Job. The default is "Any". "All" is useful when you need to broadcast an action and have it complied with on a large scale. For example, broadcasting anti-virus updates. This field is disabled when an Execution Queue is specified.

Read Only: This checkbox, when enabled, means the job's properties cannot be changed. You must have "Write" access permission to the Job's object to set this feature. To clear the read-only attribute, simply uncheck the box.

6.5.3.2 New Job – Scheduling



New Job on Queue 'quick_genq', Job Scheduler 'ASCI52'

[✓] Date/Time Trigger

Non-Recurring [● Immediately] ○ Deferred until 8/30/00 06:09 PM

Recurring (Template) ○ Interval 0 Days 0 Hrs 0 Mins ○ Schedule

☐ and Run Immediately

Calendar [None]

☐ Use Next Business Day

Time Zone to use Job Scheduler

☐ Event Trigger (Template)

Type	Namespace	Action

☐ ActiveBatch Event Triggers (Template)

☐ Run Once Only (Template)

☐ Hold on Submit

Figure 82 - New Job – Scheduling

This tab provides the ability to indicate when and how the job is to be triggered for execution.

Triggers: This section allows you to indicate what actions should trigger execution of this job. Three (3) general types of triggers are available: Date/Time, Event and ActiveBatch Event. Note:

You cannot mix recurring and non-recurring type triggers. For example, if you enable Date/Time Trigger with Immediate execution trigger, ActiveBatch will prevent you from also enabling the Event Trigger section. Remember that a recurring or potentially recurring job is a template job.

Date/Time Trigger: This checkbox, when enabled, allows a Job to be scheduled to date/time and/or time specifications. Two types of date/time triggers may be used: Non-Recurring (one-time) and Recurring.

Non-Recurring: This section refers to jobs that are scheduled using this trigger just once. Two types of non-recurring triggers are available.

Immediate: When selected the job is to be executed immediately.

Deferred until: When selected the job is to wait until the date and time specified before executing.

Recurring: This section refers to jobs that are scheduled on a periodic basis. Two types of recurring triggers are available. The checkbox **and Run Immediately** allows you to combine the best of both worlds.

Interval: This trigger, when selected, allows you to enter an interval expressed in days, hours and minutes. This “interval” is added to the starting execution time and forms the next time the job is to be scheduled for execution. For example, let’s say the time is now 11:00am, an interval of 1 day, 1 hour and 0 minutes would result in a next scheduled execution time of tomorrow at 12:00 (noon).

Schedule: This trigger, when selected, allows you to schedule a job’s execution based on any criteria of date and time specifications. Click the *Configure* button to enter or associate a Schedule with this Job. Section 6.5.4 provides much more information concerning Schedules.

Exclusion List: This button allows you to edit the date(s) and/or times that the job shouldn’t run and thus form an exclusion list. You can add, remove or update these entries. Read Section 7.3.3.1.5 for more information.

Calendar: This drop down list contains all the business calendars that have been created on this ActiveBatch system. To associate business date processing with a job, simply specify the name of the calendar you want to use. Leaving the drop down as *[None]* means that you don’t want special business date processing. The small button to the side allows you to view the selected calendar so you can be certain it’s the one you want.

Use Next Business Day: This checkbox indicates what you want done when a date/time trigger falls on a specified “holiday”. By default, the job’s trigger is skipped until the next occurrence. Setting this checkbox means that the trigger will be honored on the next business date (even though the job would not have run otherwise).

Time Zone to use: This field indicates the time zone to use for determining the time trigger. Possible time zones are: Job Scheduler, Client (Submitter’s machine) or UTC

(Universal Time Coordinated or Greenwich Mean Time).

Event Trigger: This section deals with the creation of system and/or application event triggers. The list box summarizes all the events currently associated with the Job. See Section 6.5.6 for more information on adding, editing and removing Events.

ActiveBatch Event Triggers: When enabled, this checkbox allows use of the ActiveBatch Trigger command and other release mechanisms such as Success/Failure Batch Start and Alerts. The Trigger command allows authorized users to trigger the execution of a job via command or automation.

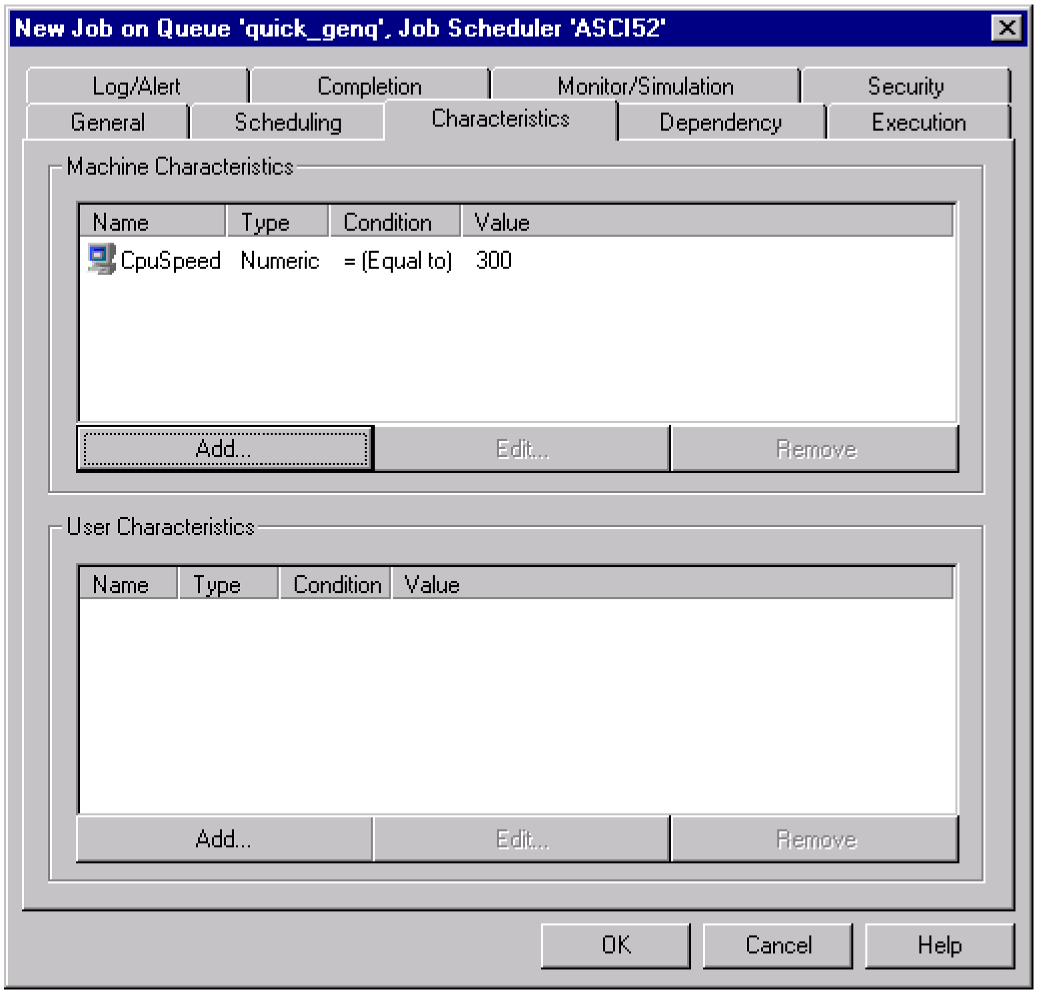
Run Once Only: When enabled, this checkbox indicates that the Job is to be run only once regardless of any date/time and/or event triggers.

Hold Job on Submit: Enabling this checkbox means the submitted job is to immediately enter a Hold state.

Note: A Job must have a minimum of one enabled trigger. By default, ABATAdmin will prompt you as to whether you would like ActiveBatch Event Triggers enabled or whether you want to edit the job triggers yourself.

Note: You may update and change a job from non-recurring to recurring, however, you may not change a job from recurring to non-recurring.

6.5.3.3 New Job – Characteristics



Name	Type	Condition	Value
CpuSpeed	Numeric	= (Equal to)	300

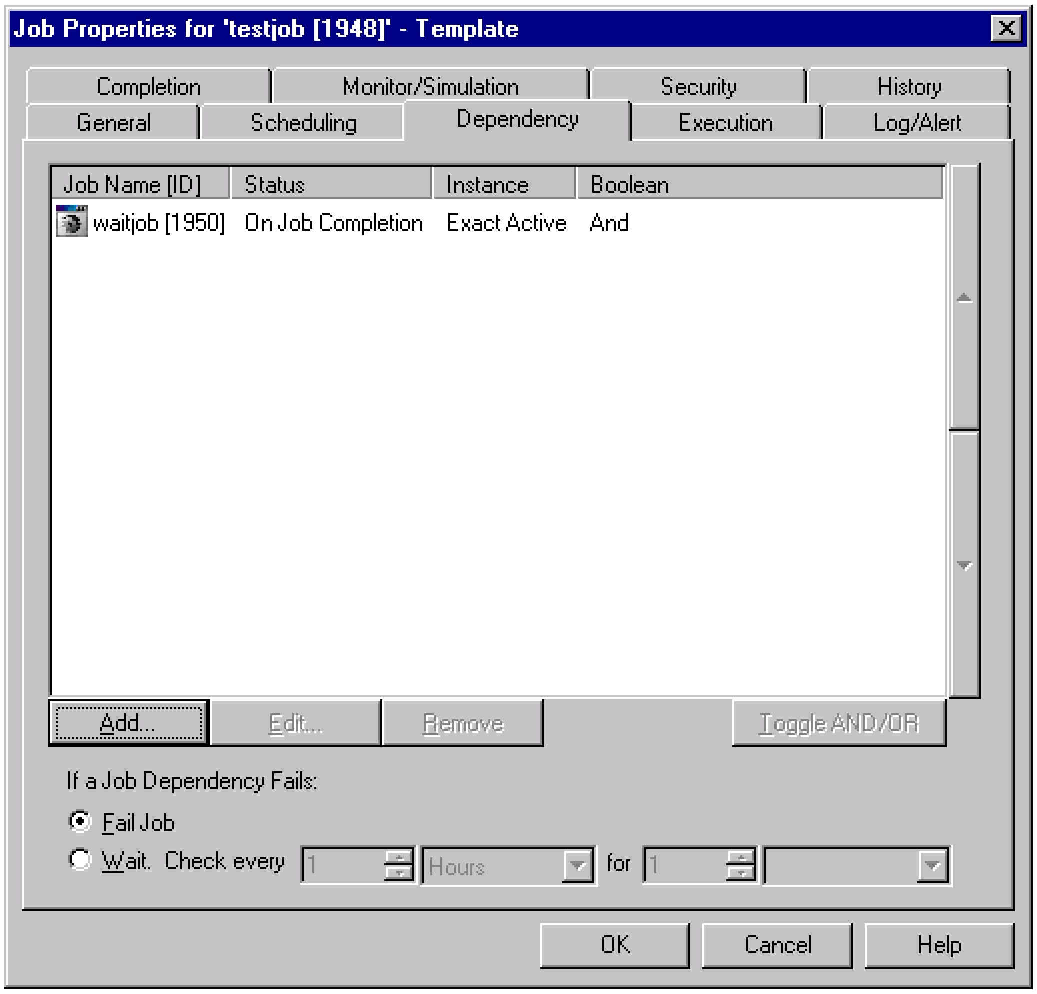
Figure 83 - New Job – Characteristics

This tab allows the job author to associate various machine and user characteristics that must “match up” in order for the proper execution queue to be selected. This tab is only applicable when a job is submitted or moved to a Generic queue.

Machine Characteristics: This list box allows you to associate certain characteristics that the execution machine must have in order for the Execution Queue(s) to be considered eligible for job execution. Existing machine characteristics associated with this job are listed. The example in Figure 83 indicates that a machine must have a CpuSpeed of at least 300Mhz to be considered eligible. To add, edit or remove associated machine characteristics, click the *Add*, *Edit* or *Remove* buttons. More information on the resulting dialogs are contained in Section 6.5.3.12.

User Characteristics: This list box allows you to associate certain user characteristics that the Execution Queue(s) must have in order to be considered eligible for job execution. Existing user characteristics associated with this job are listed. To add, edit or remove associated user characteristics, click the *Add*, *Edit* or *Remove* buttons. More information on the resulting dialogs are contained in Section 6.5.3.13.

6.5.3.4 New Job – Job Dependency


Job Properties for 'testjob [1948]' - Template

Job Name [ID]	Status	Instance	Boolean
waitjob [1950]	On Job Completion	Exact Active	And

If a Job Dependency Fails:

[✓ Fail Job]

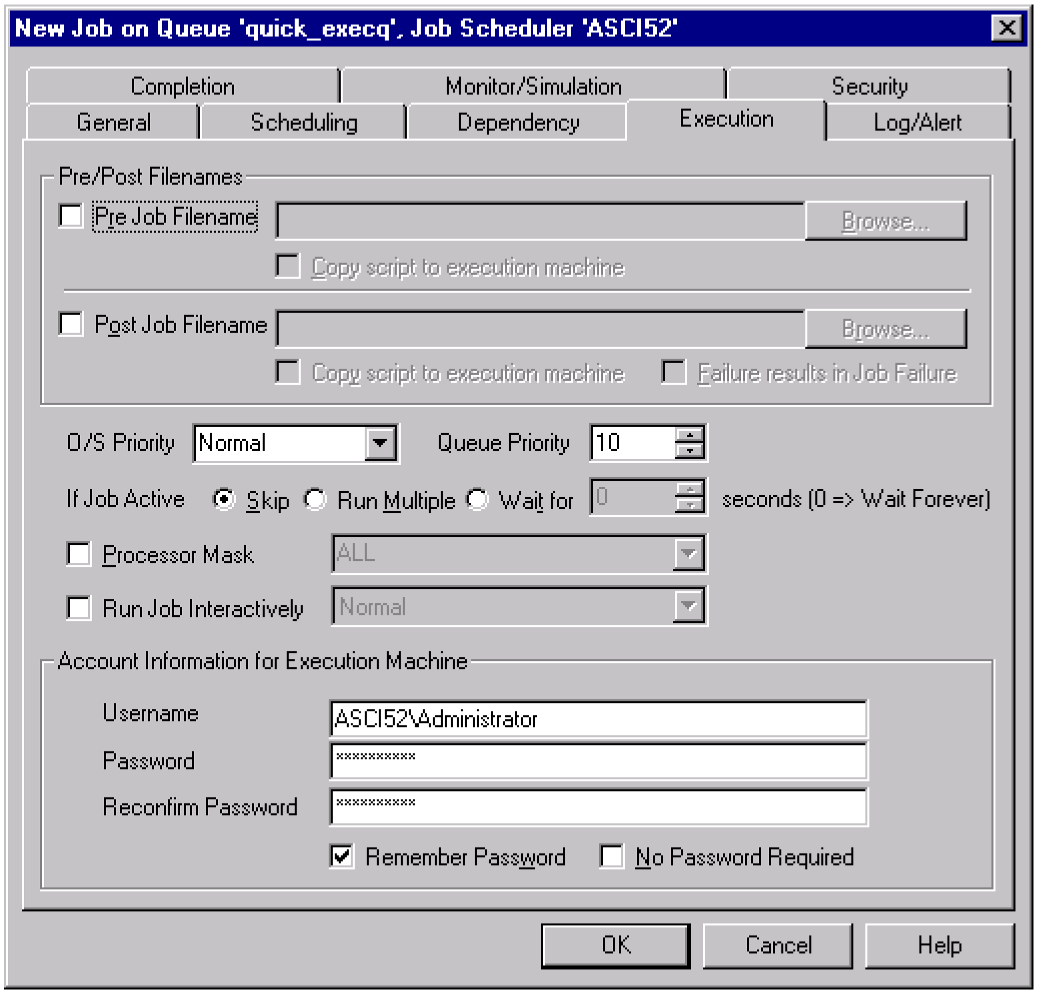
Wait. Check every 1 Hours for 1

Figure 84 - New Job -- Job Dependency Dialog

This tab deals with establishing job dependencies. A job dependency is a relationship where one job or jobs must complete (either successfully or at all) before another job can begin execution. The figure above shows a list box with each existing dependency. Dependencies can be conditional or unconditional. Dependencies can also be tied together logically using the boolean

AND/OR operator. The action buttons, *Add*, *Edit* or *Remove* can be used to add, edit or remove job dependencies. You'll also notice two large arrow buttons on the right side of the window. Those buttons allow you to change the order of any existing dependency. First, click on the dependency to select it and then click on the up or down arrow buttons to move the selected dependency to a different position. You'll also notice several fields that control what actions should be taken if one or more dependencies fail. Read Section 6.5.5 for more information.

6.5.3.5 New Job – Execution


New Job on Queue 'quick_execq', Job Scheduler 'ASCI52'

Pre/Post Filenames

☐ Pre Job Filename
☐ Copy script to execution machine

☐ Post Job Filename
☐ Copy script to execution machine ☐ Failure results in Job Failure

O/S Priority: Normal Queue Priority: 10

If Job Active: [✓ Skip] Run Multiple Wait for 0 seconds (0 => Wait Forever)

☐ Processor Mask: ALL

☐ Run Job Interactively: Normal

Account Information for Execution Machine

Username: ASCI52\Administrator
Password: **********
Reconfirm Password: **********

☑ Remember Password ☐ No Password Required

Figure 85 - New Job -- Execution Dialog

This tab provides the Job’s execution properties. ActiveBatch Job’s can consist of a Pre and Post job step surrounding the main job file (or step).

Pre/Post Filenames: This section details possible *pre* and *post* job steps.

Pre Job Filename: When enabled, the file specification entered will be executed before the main job file. This "pre" step is typically used to establish or confirm the existence of certain objects that the main step would require. For example, the presence of certain files, moving files from one machine to another, etc. The "pre" step, like the main job step, can either be a script file or a program. Unless the *Copy script…* checkbox below is enabled, the file specification must be accessible from the execution machine. You may use the *Browse* button to lookup the file specification. Table 5 shows the environmental variables (OpenVMS: logical names) that a job step may query (you cannot change any of the values).

Variable Name	Description	Job Step
ABAT_JOBID	The unique job id.	All
ABAT_JOBNAME	Job Name	All
ABAT_JOBSCHED	The machine name of the Job Scheduler.	All
ABAT_CLIENTMACHINE	The submitting user's machine name	All
ABAT_QUEUE	The queue name this job is running on.	All
ABAT_JOBFILE	The file specification of the main job file.	All
ABAT_JOBMODE	The job's mode (NORMAL | SIMULATION)	All
ABAT_CHECKPOINT	Checkpoint (YES | NO)	Main/Post
ABAT_CHECKPOINTVAL	Checkpoint value (if ABAT_CHECKPOINT true)	Main/Post
ABAT_CHECKPOINTTIME	Last Checkpoint Date/Time	Main/Post
ABAT_JOBEXITSEV	Job Exit Severity (SUCCESS | FAILURE)	Post
ABAT_JOBEXITCODE	Job Exit Code (decimal)	Post
ABAT_RESTART	Job Restart Mode (YES | NO)	All

Table 5 - Environment Variables for Job Steps

These variables allow the steps to perform whatever processing may be required including the use of COM Automation to retrieve additional information about the running job. A "pre" job step failure results in termination of the job with a "Completed(Failure)" state.

Copy script to execution machine: This checkbox controls whether the pre job step file specification is considered local or accessible from the client and should be copied to the Job Scheduler machine and then ultimately to the Execution machine. The file size must be less than the System policy *CopyScriptMaxSize*.

Post Job Filename: When enabled, the file specification entered will *always* be executed after the main job file. This “post” step is typically used to either clean up any temporary or other files that are no longer necessary and/or move files to other machines. The main job step’s exit status is also provided to the “post” step. The “post” step, like the main job step, can either be a script file or a program. Unless the *Copy script…* checkbox below is enabled, the file specification must be accessible from the execution machine. You may use the *Browse* button to lookup the file specification. The variables mentioned in Table 5 allow the “post” step to determine how the main job step did and to then perform success or failure processing. The “Restart” indication is also useful in that many jobs may require new copies of various files before the main step executes. With pre and post step processing such determinations no longer need to be made by the main job step. The use of COM Automation to retrieve additional information about the running job may also be used. The checkbox **Failure results in Job Failure** allows you to indicate whether the failure of the “post” step should result in failure of the entire job.

Copy script to execution machine: This checkbox controls whether the post job step file specification is considered local or accessible from the client and should be copied to the Job Scheduler machine and then ultimately to the Execution machine. The file size must be less than the System policy *CopyScriptMaxSize*.

Base Priority: This field indicates the Operating System priority that should be assigned to the Job. For Windows 2000/NT, values of Low, BelowNormal, Normal, AboveNormal, High and Realtime are used. For non-Windows 2000/NT systems, a numeric value from 1 to 32 may be used. The default for this field is taken from the Queue’s *Base Priority* property.

Queue Priority: This field indicates the priority of the Job within the ActiveBatch system. The priority is used when queuing a job for submission to a named Queue. Values from 1 to 100 are acceptable. The higher the value, the higher the priority. The default for this field is taken from the Queue’s *Queue Priority* property.

If Job Active: This field indicates what action should be performed if an instance of this Job is already running.

Skip: If selected, skip execution of this instance. In other words, only one instance can run.

Run Multiple: If selected, run the instance. This means multiple instances of the same job can be running.

Wait for: If selected, this instance is to wait, for a specified time (in seconds), before this run is skipped. (While you can specify zero, we recommend you specify an actual value).

Run Job Interactively: If enabled, Windows 2000/NT Execution Agents will run the job on the user's desktop. Three (3) modes are available: Normal, Maximized and Minimized. Typically, jobs are not run interactively but in background mode. Note: Running an interactive job on a desktop does pose a security risk since the desktop user will have possible access to interactive job.

Processor Mask: If enabled and the execution machine is multiprocessor capable, the mask is used to indicate which processors this job can use. The processor mask only applies to the main job step. The keyword ALL means all processors should be used (default). To enter a specific processor(s) specify the CPU number as a comma or dashed based entry. For example, 0 means CPU 0 should be assigned. "0,2,5-7" means CPUs 0, 2, 5, 6, 7 should be used. This property may not be supported by all operating systems so please read the Execution Agent Release Notes for applicability.

Account Information for Execution Machine: This section is where the user must enter authorization credentials for the execution machine where the job is to execute.

Username: This mandatory field contains the username account. By default, your currently connected username (and domain) are used[3]. If this username is not applicable to the specified execution machine, specify a username that is acceptable. This field is case sensitive only if the execution machine considers the field case sensitive.

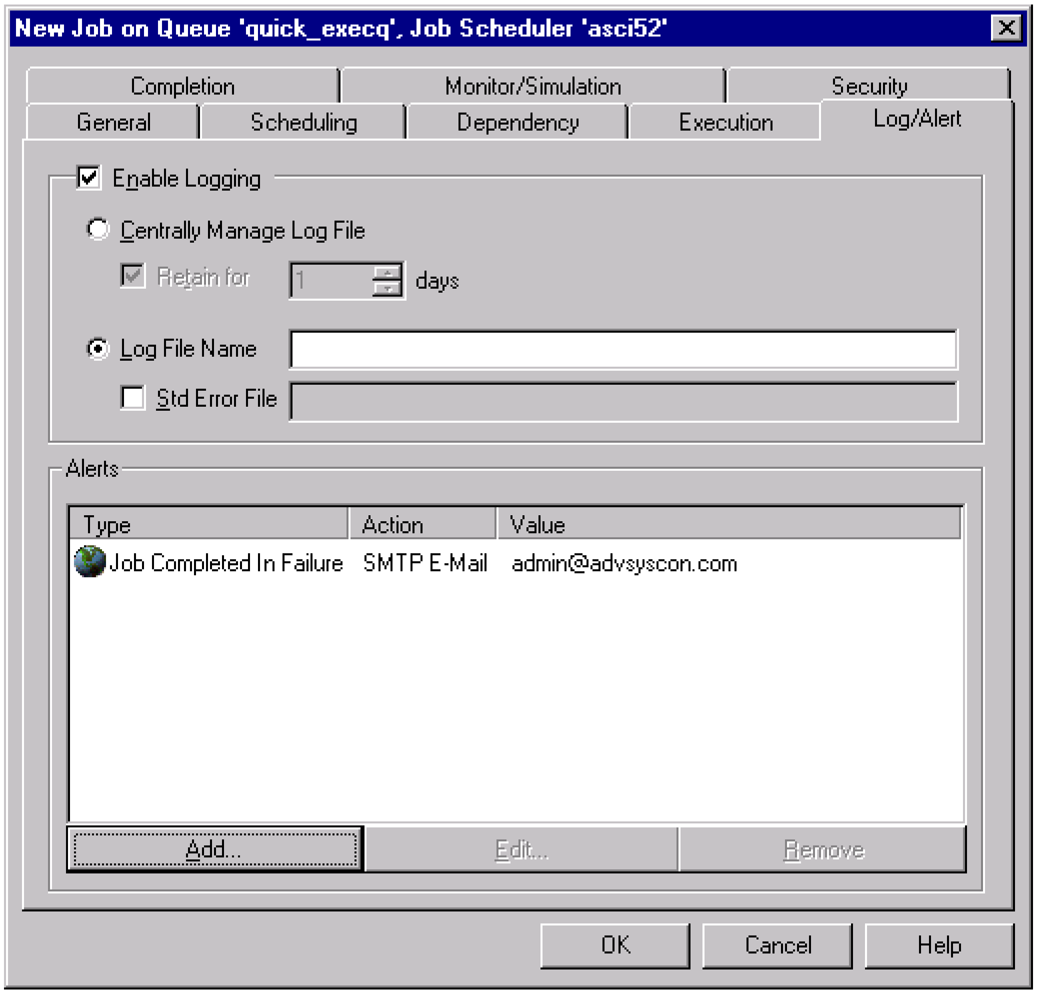
Password: This field contains the corresponding password to the above username account. The field is not displayable although a series of asterisks appears as you enter data. This field is case sensitive only if the execution machine considers the field case sensitive. You must enter data into this field unless the **No Password Required** checkbox is enabled or your password had been previously saved.

Reconfirm Password: This field uses the same attributes as the Password field and is present to reconfirm the unseen password. If a mismatch occurs, you will be immediately notified.

Remember Password: This checkbox allows your authorization credentials to be stored using the same facility as Windows. If this is the first time you have entered your credentials with this checkbox enabled, your security information is saved. On subsequent uses, the clicked checkbox means that ActiveBatch will retrieve and use the saved credentials. If you don't want to use those saved credentials anymore, simply unclick the checkbox. This will cause the currently saved credentials to be removed and the ones specified will be used.

[3] Typically your "connected" username will be your "logged on" username, however, ActiveBatch does allow you to connect to a Job Scheduler with different security credentials.

6.5.3.6 New Job – Logging/Alerts



[✓] Enable Logging

○ Centrally Manage Log File

[✓] Retain for 1 days

● Log File Name

☐ Std Error File

Alerts

Type	Action	Value
Job Completed In Failure	SMTP E-Mail	admin@advsyscon.com

Figure 86 - New Job – Log/Alerts

This tab is used to control the capture of standard output and standard error for the job and to also associate alerts.

Enable Logging: Clicking on the checkbox enables the possible capture of standard output and/or error. If the checkbox is not enabled, standard output and error are not captured (no files are created).

Centrally Manage Log File: Clicking on the checkbox enables this feature. Centrally managed log files means that ActiveBatch will handle the complete specification of the log and/or error files and the directories they are created in. You can view the log file(s) from the Job History tab as well as determine where the files are located.

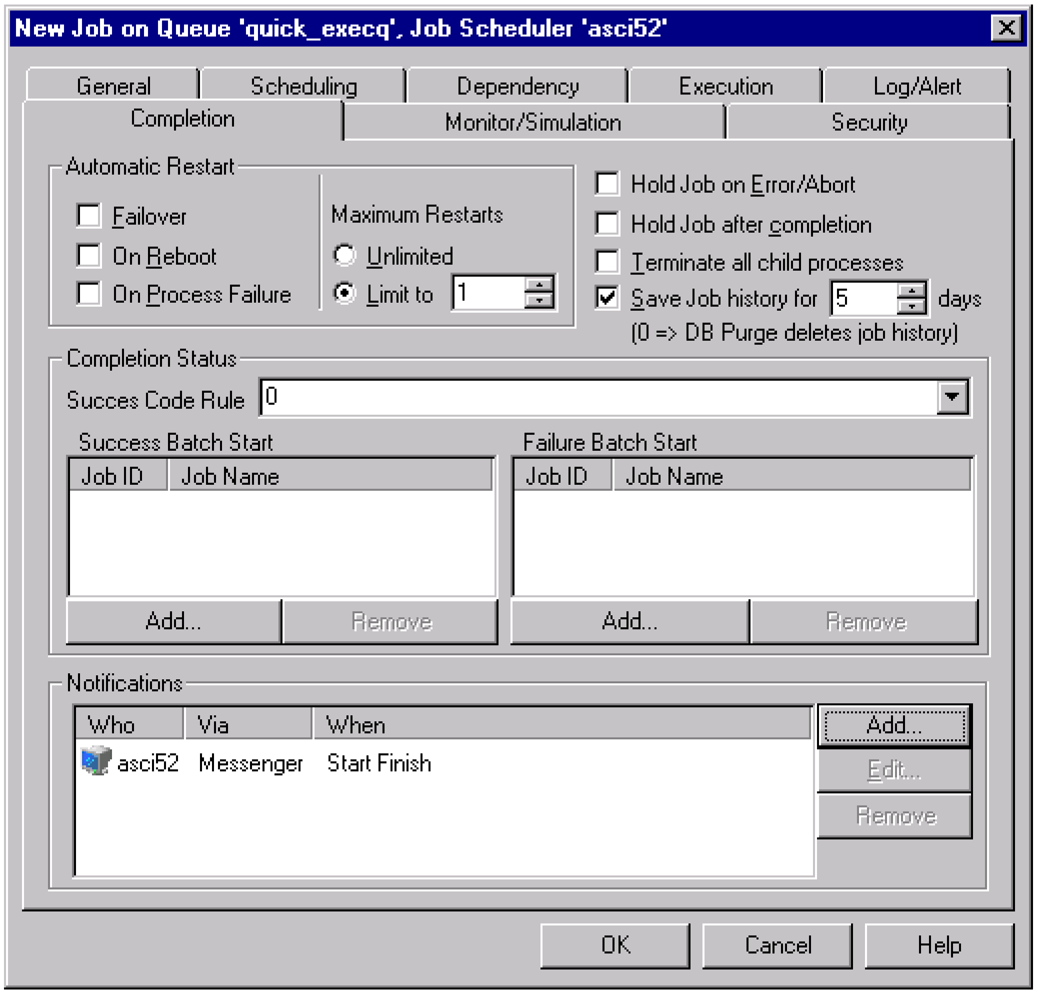
Retain for: This field indicates whether you want the centrally managed log file(s) to be retained for a specified period of time. Clicking the checkbox means you want the log file(s) to be retained after the job completes. The period of time is measured in days. The default is one (1) day. A value from 1 to 366 is acceptable. Not enabling this field means you want the log file(s) to be deleted after the job has completed.

Log File Name: Clicking on the checkbox means you want the standard output file to be created. This field and centrally managed log files are mutually exclusive. For non-centrally managed files, the file specification entered must be accessible from the execution machine point-of-view. Please note that for recurring jobs, the log file specification will also have the time and date appended to the file spec for uniqueness purposes. (This is to avoid a conflict where multiple instances of the job have executed and would otherwise overwrite the log file). The default varies depending on whether the batch file was copied to the execution system. If the batch file was copied to the execution machine, the log file is created in the job's working directory with an extension of .LOG. If the batch file was not copied to the execution machine, the log file is created in the same directory as the batch file with an extension of .LOG.

Std. Error File: Clicking on the checkbox means that you want the standard error stream to be captured. A file specification must be entered if this checkbox is enabled. If omitted, a batch job's standard error is directed to the same log file as standard output.

Alerts: This area lists all alerts associated with this job. You may add, edit or remove alerts by clicking the appropriate buttons. See Section 6.5.9 for more information on setting job alerts.

6.5.3.7 New Job – Completion


New Job on Queue 'quick_execq', Job Scheduler 'asci52'

Automatic Restart
- [ ] Failover
- [ ] On Reboot
- [ ] On Process Failure

Maximum Restarts
- ○ Unlimited
- ● Limit to: 1

- [ ] Hold Job on Error/Abort
- [ ] Hold Job after completion
- [ ] Terminate all child processes
- [✓] Save Job history for 5 days
(0 => DB Purge deletes job history)

Completion Status
Succes Code Rule: 0

Success Batch Start

Job ID	Job Name

Failure Batch Start

Job ID	Job Name

Notifications

Who	Via	When
asci52	Messenger	Start Finish

Figure 87 - New Job – Completion

This tab provides properties that concern the completion phase of a Job. This includes what to do if the execution machine crashes, interpreting success or failure exit codes, notifications and triggering other batch jobs.

Automatic Restart: This section determines when and if ActiveBatch is to restart a Job. Typically restart concerns the job's ability to be automatically restarted in the event of a machine failure (usually the execution machine).

> **Failover:** Enabling this checkbox allows an executing job, on an execution machine failure, to failover back to the original Generic Queue it was submitted to for re-dispatch. This is particularly useful for Microsoft Cluster Server or any fault tolerant cluster systems. *Failover* is used only for Generic Queues. If *Failover* and *On Reboot* are both enabled, *Failover* is used.
>
> **On Reboot:** Enabling this checkbox indicates that a job that runs on an execution machine that fails is to be restarted and to wait for the machine to become available. This feature is available for both Generic and Execution Queues. If both *Failover* and *On Reboot* are specified, *Failover* is used if the original Queue is Generic otherwise *On Reboot* is used for an Execution Queue.
>
> **On Process Failure:** This checkbox should be enabled with caution. Enabling this feature means the job will be restarted if the job fails. Where as *Failover* and *On Reboot* concern machine failures, *On Process Failure* concerns the job itself. Operator aborted jobs are not subject to automatic restart.
>
> **Maximum Restarts:** The radio buttons allow for an unlimited number of restarts (not really recommended from a practical point-of-view) or a specific number of restarts (the acceptable value range is 1 to 999). The maximum number of restarts controls the total number of restarts attempted for *this* instance.

Hold Job on Error/Abort: Enabling this checkbox means that should a Job terminate in error or be terminated, the job should be placed in a Hold state. For non-recurring jobs the single instance is held. For recurring jobs the instance becomes "history" and the template is held.

Hold Job after Completion: Enabling this checkbox means that the job should be held on completion regardless of its exit status. This affects both single instance and template jobs.

Terminate all child processes: Enabling this checkbox means that all processes created by the job are to be terminated when the job completes. For Windows NT V4 systems, the "parent" process ID is used to track child processes. For Windows 2000 systems, the job is created as an NT Job object and all children can be tracked. For UNIX systems, parent/child processes are part of the operating system. For OpenVMS systems, sub-processes are terminated regardless of the setting of this checkbox and detached processes are never terminated.

Save Job History for: This checkbox, when enabled, indicates how long a completed job's history is to be retained within ActiveBatch's database. If the feature is enabled (highly recommended), a value of 0 indicates that the next scheduled running of DBPurge should delete the job's history (and log files). The value specified is defined in days and an acceptable range is 0 through 366. If the checkbox is not enabled, the job's history will be deleted upon the next scheduled running of the DBPurge facility.

Completion Status: This section concerns how a job's exit status is to be interpreted.

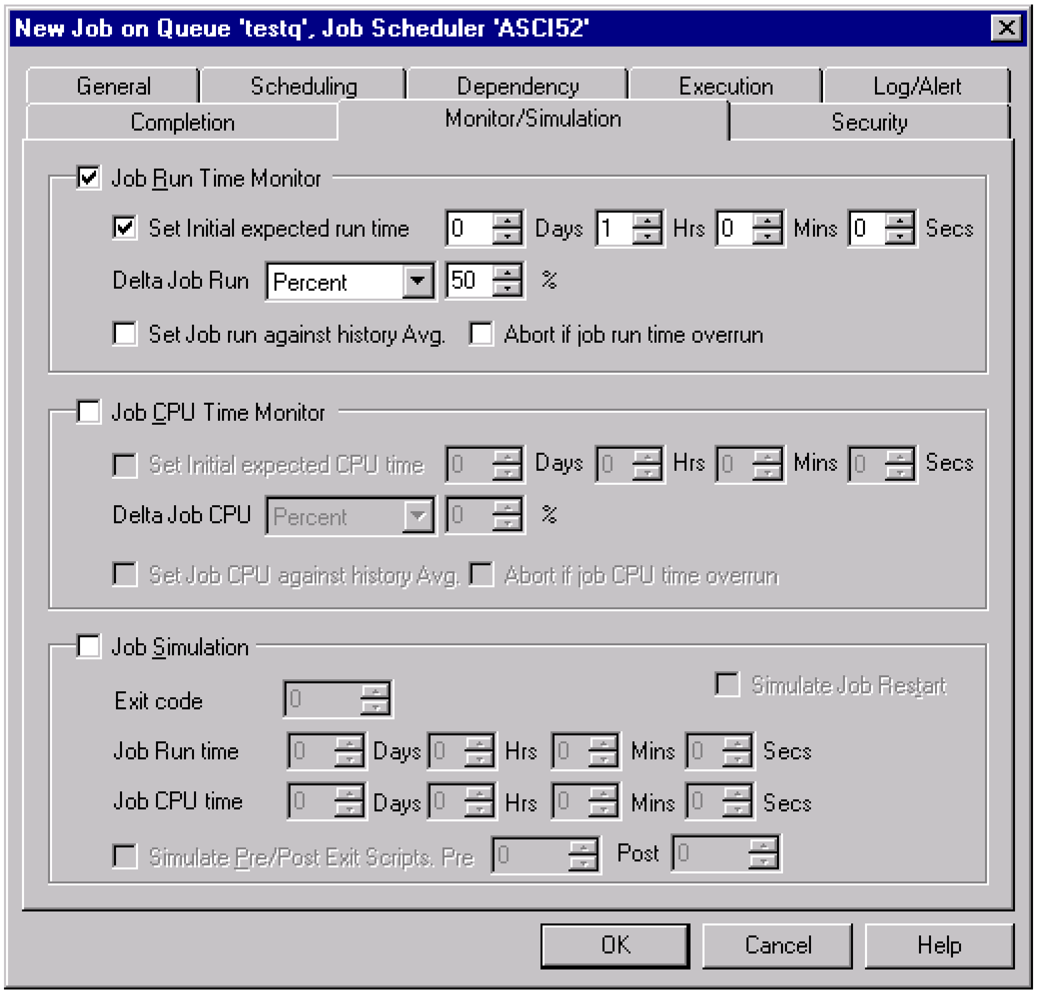
> **Success Code Rule:** This field can contain keywords and numeric (decimal) exit codes all of which are interpreted as success. A range can be specified as "n-n" (for example, 0-5 means zero through 5). Several keywords are also supported: *NTMsg*, *Odd*, *Even* and *LZero* (negative). *NTMsg* indicates that an NT Message Code is passed back as an exit status. The upper two (2) bits indicate the severity. XLNT uses this type of success rule. *Odd* and *Even* indicate status codes which are either odd or even numbers. OpenVMS uses an odd status as a success indication. Multiple codes can be specified by separating whole positive numbers and ranges with commas (for example: 0,5-10,20). This field offers a simple pulldown control with zero, NTMSG, LZERO, ODD and EVEN as quick selection choices.

Success Batch Start: This list box indicates which job(s) are to be triggered for execution if this job completes successfully. The job you select must have ActiveBatch Event trigger enabled. You can add, edit or remove batch start triggers by clicking on the appropriate buttons. More information on this facility can be found in Section 6.5.7. Please note that the batch jobs to be triggered must exist and be defined within ActiveBatch.

Failure Batch Start: This list box indicates which job(s) are to be triggered for execution if this job fails. The job you select must have ActiveBatch Event trigger enabled. You can add, edit or remove batch start triggers by clicking on the appropriate buttons. More information on this facility can be found in Section 6.5.7. Please note that the batch jobs to be triggered must exist and be defined within ActiveBatch.

Notifications: This list box indicates all notification actions the job should execute on job completion. You can add, edit or remove these actions by clicking on the appropriate button. More information on this facility can be found in Section 6.5.8.

6.5.3.8 New Job – Monitor/Simulation


New Job on Queue 'testq', Job Scheduler 'ASCI52'

☑ Job Run Time Monitor

☑ Set Initial expected run time 0 Days 1 Hrs 0 Mins 0 Secs

Delta Job Run Percent 50 %

☐ Set Job run against history Avg. ☐ Abort if job run time overrun

☐ Job CPU Time Monitor

☐ Set Initial expected CPU time 0 Days 0 Hrs 0 Mins 0 Secs

Delta Job CPU Percent 0 %

☐ Set Job CPU against history Avg. ☐ Abort if job CPU time overrun

☐ Job Simulation

Exit code 0 ☐ Simulate Job Restart

Job Run time 0 Days 0 Hrs 0 Mins 0 Secs

Job CPU time 0 Days 0 Hrs 0 Mins 0 Secs

☐ Simulate Pre/Post Exit Scripts. Pre 0 Post 0

Figure 88 - New Job -- Monitor/Simulation Dialog

This tab controls the ability for ActiveBatch to proactively monitor a job's progress and possibly terminate a job that isn't running as expected. The example above shows an expected time of one (1) hour with an allowable under/over delta of fifty (50) percent. You can also simulate the various ActiveBatch actions and test them to determine whether they are working as you expect.

Job Run Time Monitor: This section, when enabled, allows ActiveBatch to examine the job's expected elapsed time and determine whether the job is operating as expected.

Set Initial expected run time: If selected, you enter a days, hours, minutes and seconds time which represents the expected job's elapsed run time.

Delta Job Run: Three (3) fields make up this property. The Delta Job property allows you to indicate an acceptable under/overrun on elapsed time. You can enter the delta time as either a percentage or as a time spec (days, hours, minutes and seconds).

Abort if job run time overrun: If enabled, ActiveBatch will abort the executing job on an overrun condition. An overrun condition occurs when the expected run time plus the delta time values (pct or time) are exceeded.

Set Job run against history avg: If enabled, ActiveBatch will refine the job's elapsed time by using the actual run time and average it against previous acceptable runs.

Reset: This button is available when modifying an existing job. Click this button to reset any historical averaging values that ActiveBatch is retaining.

Job CPU Time Monitor: This section, when enabled, allows ActiveBatch to examine the job's expected CPU time and determine whether the job is operating as expected.

Set Initial expected run time: If selected, you enter a days, hours, minutes and seconds time which represents the expected job's CPU run time.

Delta Job CPU: Three (3) fields make up this property. The Delta Job property allows you to indicate an acceptable under/overrun on CPU time. You can enter the delta time as either a percentage or as a time spec (days, hours, minutes and seconds).

Abort if job CPU time overrun: If enabled, ActiveBatch will abort the executing job on an overrun condition. An overrun condition occurs when the CPU run time plus the delta time values (pct or time) are exceeded.

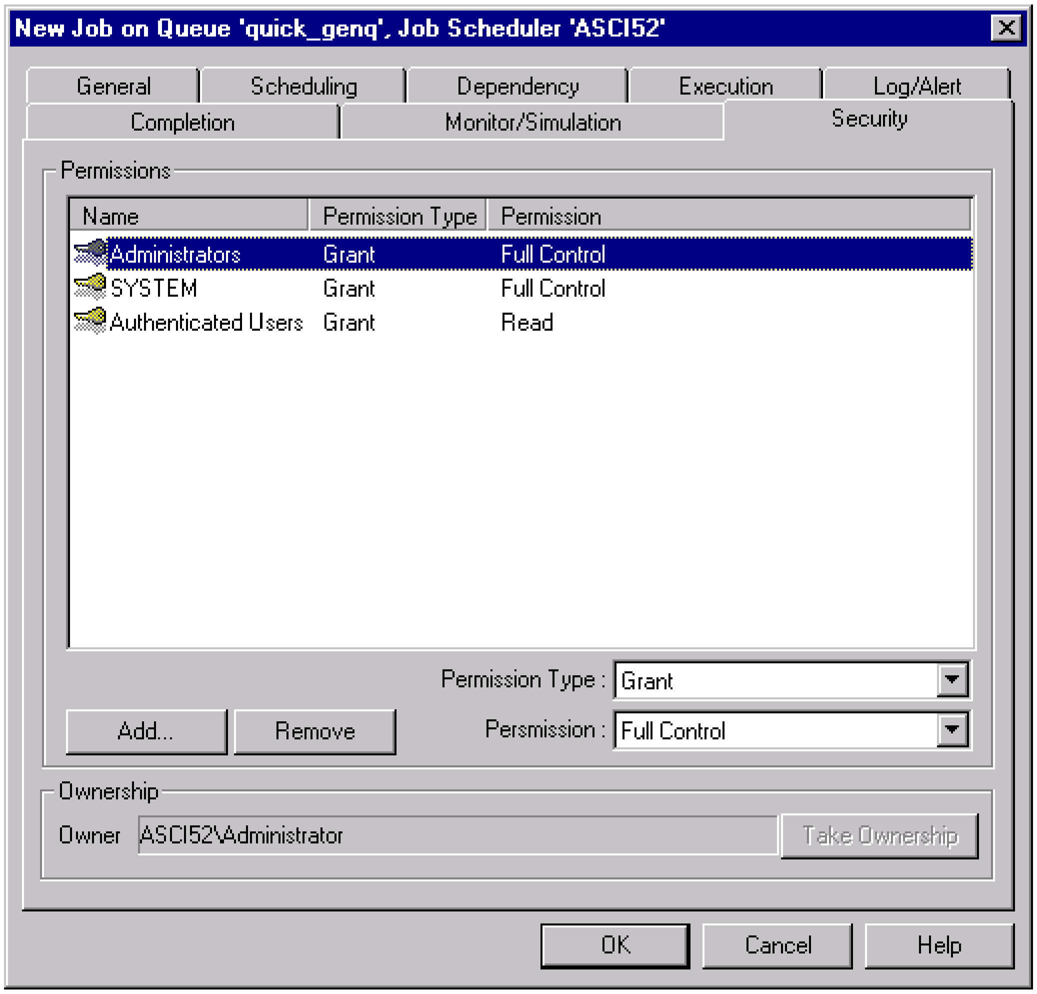
Set Job CPU against history avg: If enabled, ActiveBatch will refine the job's CPU time by using the actual CPU run time and average it against previous acceptable runs.

Reset: This button is available when modifying an existing job. Click this button to reset any historical averaging values that ActiveBatch is retaining.

Job Simulation: This section concerns the ActiveBatch simulation facility. This facility allows you to test various aspects of the job you've defined without actually running the job. This checkbox, when selected, enables this facility.

> **Simulation Exit Code:** You may enter a decimal number which will be interpreted by ActiveBatch as the exit code of the main job step.
>
> **Job run time for simulation:** You may enter a time period which will be used for elapsed time monitoring purposes (if monitoring is enabled).
>
> **Job CPU time for simulation:** You may enter a time period which will be used for CPU time monitoring purposes (if monitoring is enabled).
>
> **Simulate Pre/Post Script Exit Codes:** You may enter a decimal value which will be interpreted as an exit code for the "pre" job step.
>
> **Post Simulation:** You may enter a decimal value that will be interpreted as an exit code for "post" job step.
>
> **Simulate Job Restart:** This checkbox, if enabled, allows you to simulate a job restart condition for proper testing of pre, main and post job steps.

6.5.3.9 New Job – Security



Name	Permission Type	Permission
Administrators	Grant	Full Control
SYSTEM	Grant	Full Control
Authenticated Users	Grant	Read

Permission Type : Grant

Persmission : Full Control

Owner ASCI52\Administrator

Figure 89 - New Job -- Security Dialog

This tab controls the security attributes associated with this Job object. The ActiveBatch Administrator may select Job Policies for an ActiveBatch installation and these policies extend to the security system. Therefore when you create a Job you may notice that several permissions have already been associated with the object.

To add, edit or remove security access permissions requires the user to have "Executive" access to the object.

This table lists all security access permissions using Windows 2000/NT conventions. To edit an existing account name, select the listed account and change the permission using the **Permission:** list box. You may select one of the following access permissions:

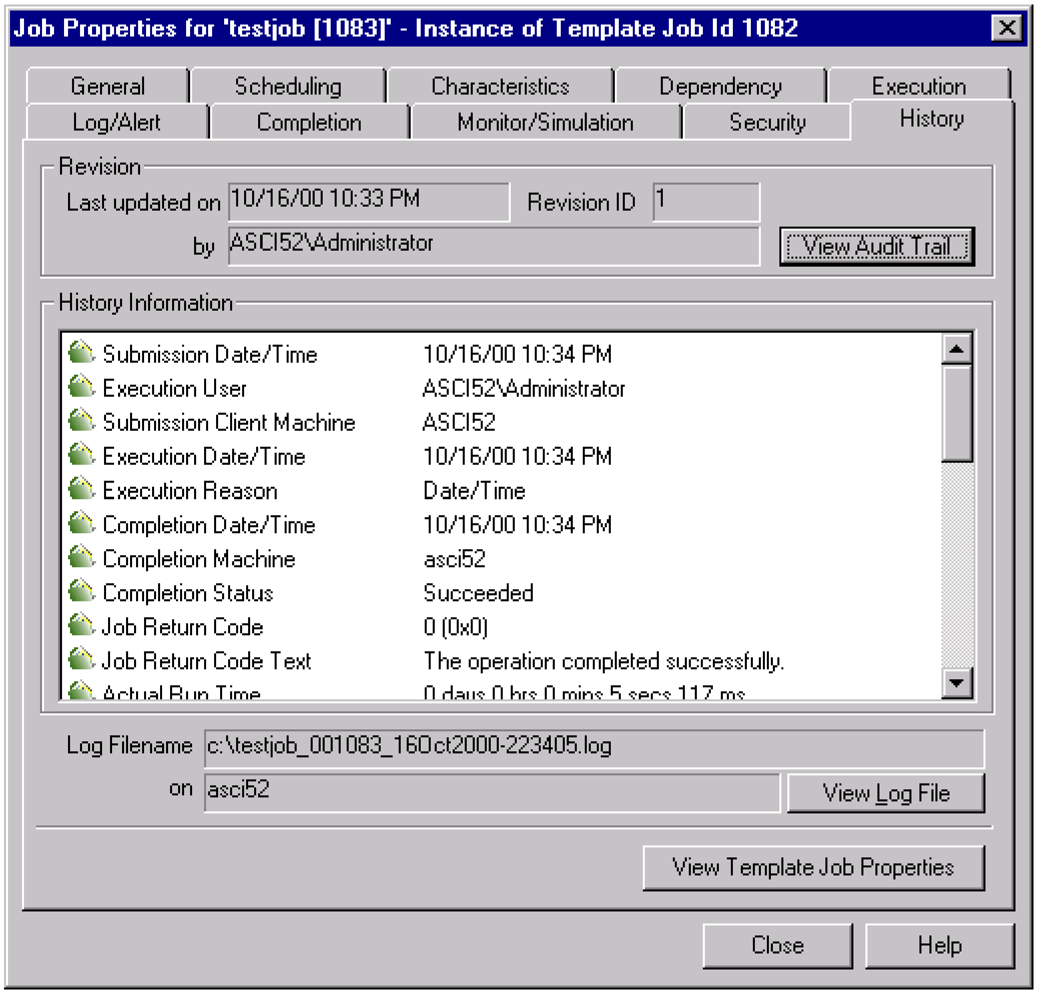
Access	Description
Read	User is allowed to read the job's properties
Submit	User is allowed to submit the job for execution
Write	User is allowed to update the job properties (except security)
Manage	User is allowed to issue job operations
Delete	User is allowed to delete the job
Executive	User is allowed to update the job's security as well as take ownership
Trigger	User may trigger job
Trigger and Change Queue	User may trigger job and direct job to execute on a specified Queue.
Trigger and Change Parameters	User may trigger job and specify new input parameters to job
Full Control	All of the above permissions
No Access	No access permissions (Grant Only)

Table 6 - Job Security Access

To add a new access permission, click the *Add* button and follow the dialog as discussed in Section 6.5.3.11. To remove an existing account name, select the listed account and click the *Remove* button. To change an existing access permission, select the account and then select a new **Permission Type** (either Grant or Deny access) and a new **Permission** (one of the access permissions listed in Table 6.

The owner of a job is the user who first creates the Job. The owner of a job is implicitly granted *Full Control* access permission. To change ownership, click the *Take Ownership* button and confirm the resulting dialog.

6.5.3.10 Job – Job History Dialog



Revision

Last updated on 10/16/00 10:33 PM   Revision ID 1

by ASCI52\Administrator

History Information

Submission Date/Time	10/16/00 10:34 PM
Execution User	ASCI52\Administrator
Submission Client Machine	ASCI52
Execution Date/Time	10/16/00 10:34 PM
Execution Reason	Date/Time
Completion Date/Time	10/16/00 10:34 PM
Completion Machine	asci52
Completion Status	Succeeded
Job Return Code	0 (0x0)
Job Return Code Text	The operation completed successfully.
Actual Run Time	0 days 0 hrs 0 mins 5 secs 117 ms

Log Filename c:\testjob_001083_16Oct2000-223405.log

on asci52

Figure 90 - Job History (Instance)

The Job History tab appears when a job has already been defined and submitted to the system. This tab provides a lot of useful information about the job as it moves from pending to executing to completed. This tab is a good place to begin when you're not sure why a job failed or, for that matter, failed to even execute. The Job History dialog box is separated into two areas: Revision and Job History.

Revision: This area indicates the number of revisions made to this job and the date/time and user who made the last revision.

View Audit Trail: This button provides a complete view of all modifications and ActiveBatch interactions with the job. Clicking this button causes the Audit Trail window to appear (Figure 95).

Job History: This list box provides relevant information concerning the running of the job itself.

Submitted Date/Time: The date and time that the job was initially submitted for execution.

Execution User: The user under whose security context the job is run.

Submission Client Machine: The machine name of the user who submitted this job.

Execution Date/Time: The date and time the job began execution.

Execution Reason: Why the job began execution.

Completed Date/Time: The date and time the job completed execution.

Completion Machine: The name of the last machine that ran the job.

Completion Status: SUCCESS or FAILURE indication for the job.

Job Return Code: This field contains the actual job exit code (decimal).

Job Return Code Text: ActiveBatch attempts to interpret the "job return code" as an NT Message Code (so you may need to ignore this field). Read Section 6.1.7 for information on how to enable or disable this feature.

Actual Run Time: Actual elapsed time of the job.

Actual CPU Time: Actual CPU time of the job.

Aborted: YES or NO indicating whether the job was manually terminated.

Restarted: This YES or NO field indicates whether the job was restarted.

Restart Count: Number of times the job was restarted.

Restart Date/Time: If this job had been restarted, this indicates the last date and time the restart took place.

Failed Over: YES or NO indicating whether this job had been failed over to another machine.

Checkpoint Restart Enabled: YES or NO indicating whether a valid checkpoint has been taken for this job.

Checkpoint Restart Value: The value of the checkpoint (if enabled).

Last Checkpoint Date/Time: The last date/time that a checkpoint was taken.

Pre Job Return Code: Actual pre-job exit code.

Post Job Return Code: Actual post-job exit code.

Submission Queue: Queue name job was originally submitted to.

Submission User: Username who originally submitted this job.

Trigger by User: Username who issued Trigger command.

Job Execution Step: The job steps selected for this job (pre, main, post).

Job Execution Steps Completed: The job steps that have already been completed.

Simulated: YES or NO indicating whether this job ran as a result of a simulation.

Event Trigger ID: If the Execution Reason is "Event Trigger" then this indicates the actual event ID that triggered the job.

Event Trigger Machine: The name of the machine that triggered the event.

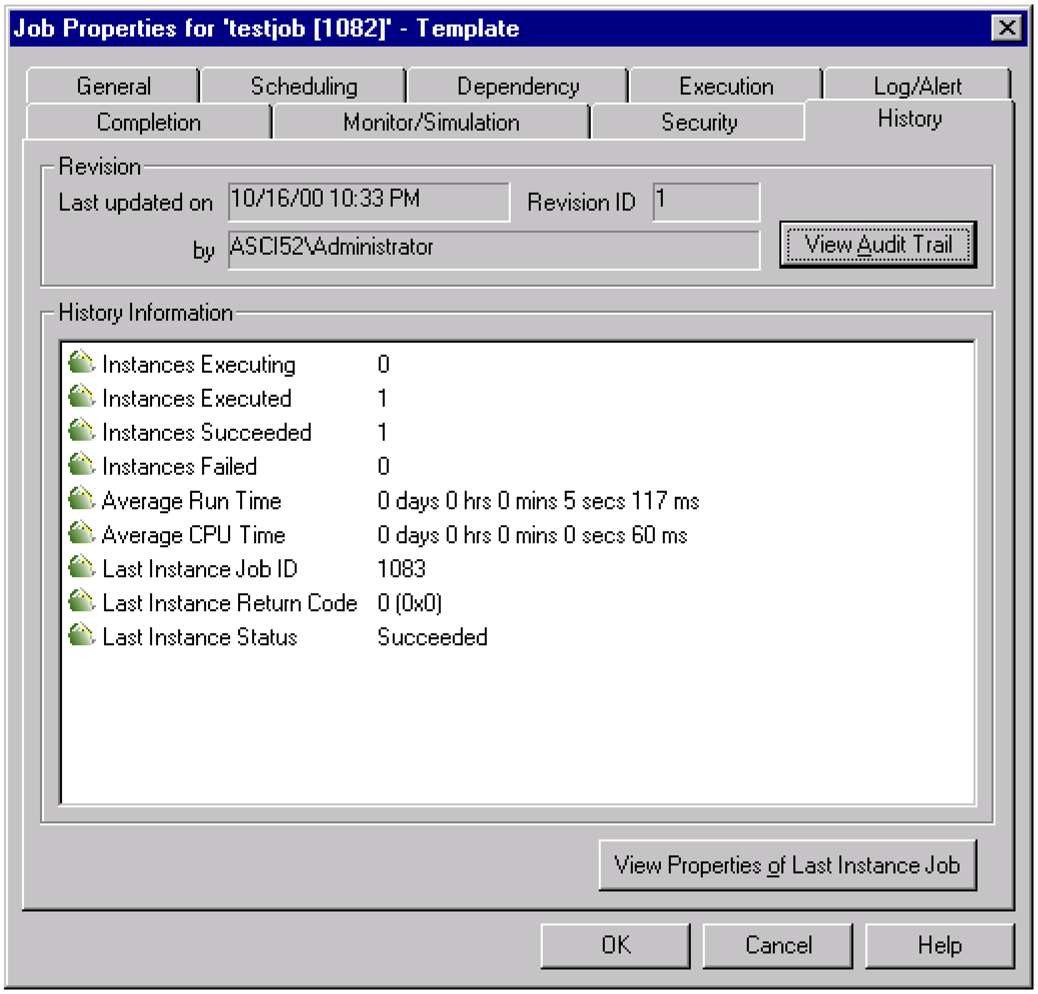
Alerted: YES or NO indicating whether the job had been the subject of any alerts.

Last Alert Date/Time: If Alerted is true, then this is the date and time of the last issued alert.

Log File: This field contains the actual location and log file name used by the batch job.

View Log File button allows you to view a Log File. The editor selected can be set by through Preferences (the default is Notepad). This button is only selected if the file can be viewed through either the Managed Log File Share (ASCI_ABATLOG) or a UNC or local path is available. More information on the ASCI_ABATLOG share is available in the section on Installation (Section 3.1.1).

View Template Job Properties button (only if an instance is being viewed) allows you to select and view the instance's template properties.



Revision

Last updated on 10/16/00 10:33 PM Revision ID 1

by ASCI52\Administrator

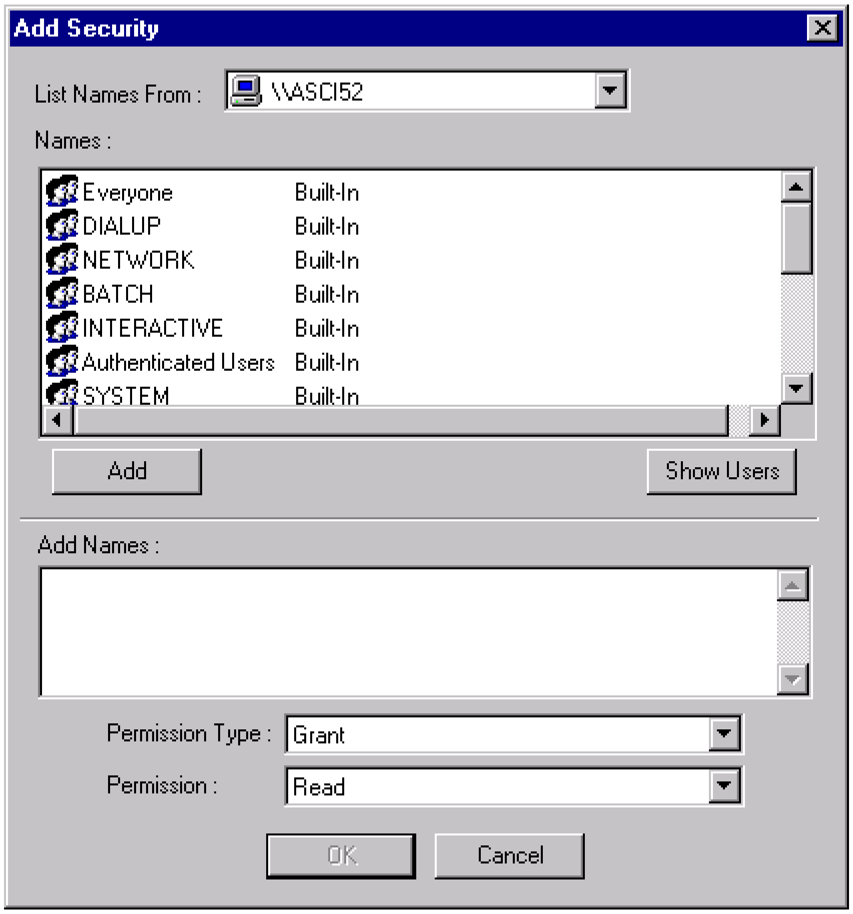
History Information

Instances Executing	0
Instances Executed	1
Instances Succeeded	1
Instances Failed	0
Average Run Time	0 days 0 hrs 0 mins 5 secs 117 ms
Average CPU Time	0 days 0 hrs 0 mins 0 secs 60 ms
Last Instance Job ID	1083
Last Instance Return Code	0 (0x0)
Last Instance Status	Succeeded

Figure 91 - Job History (Template)

The figure above shows history information for the template job. You can view the properties and history of the last instance by clicking on the button *View Properties of Last Instance Job*. This will pop up a properties display of the last instance executed.

6.5.3.11 Job – Add Security Dialog



Everyone	Built-In
DIALUP	Built-In
NETWORK	Built-In
BATCH	Built-In
INTERACTIVE	Built-In
Authenticated Users	Built-In
SYSTEM	Built-In

Permission Type : Grant

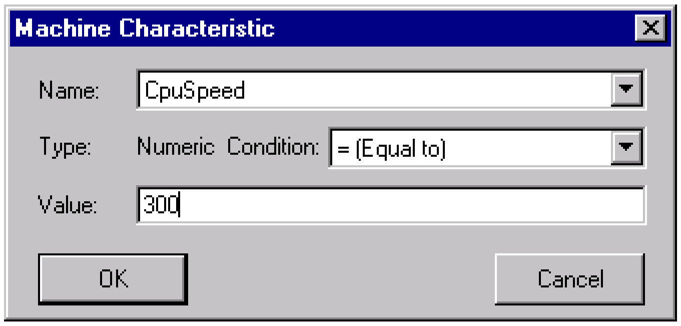
Permission : Read

Figure 92 - Job Add Security Dialog Box

The above figure is displayed when you want to add one or more accounts to a Job object. The drop down list box shows all available domain names as well as the Job Scheduler machine. The top portion indicates the groups you may select. Clicking the *SHOW USERS* button causes all the user accounts on the selected domain or machine to be displayed.

To add a user or group, simply select the user or group from the top list and click the *Add* button. That user or group will appear in the bottom box (labeled “Add Names”). You may also manually enter accounts directly into the Add Names section. To select the type of access permission, you need to click on the security access permissions drop-down box (labeled “Type of Access”). The type of access permissions available are described in Table 6.

6.5.3.12 Job – Machine Characteristics



Name: CpuSpeed
Type: Numeric Condition: = (Equal to)
Value: 300

Figure 93 - Machine Characteristic Dialog

This dialog appears when you want to add or edit a machine characteristic to a Job. Machine characteristics contain static information about the execution machine. The figure above indicates that the job is to run on a machine equal to 300mhz in CPU speed. Machine characteristics are used by ActiveBatch when a job is submitted to a Generic Queue and the proper Execution Queue (machine) needs to be selected. This allows the author of a Job to indicate what minimum runtime requirements the job has. Remember that user/machine characteristics are ignored when a job is submitted directly to an Execution Queue.

Name: This field contains a dropdown list of the machine characteristics. The following table provides the machine characteristics that are available (see the ReadMe file for any new characteristics that may have been added).

Characteristic	Description
CPU	(String) CPU Platform Type. (Example: Pentium)
CPUArch	(String) CPU Architecture. (Example: x86)
Memory	(Numeric) Physical Memory Size in MB.
CPUSpeed	(Numeric) Processor Speed in Mhz. Note: Not all operating system's expose a chip's processor speed.
OS	(String) Operating System. (Example: Windows NT)

OSVersion	(String) Operating System Version. (Example: V4)
OSServicePack	(String) Operating System Service Pack. Note: Not all operating system's use a Service Pack. (Example: 5)
OSBuildNumber	(String) Operating System Build Number. Note: Not all operating system's use a build number. (Example: 1381)
OSType	(String) Operating System Platform Type. (Example: Workstation)
ABATVersion	(String) ActiveBatch Execution Agent Version. (Example: V3)

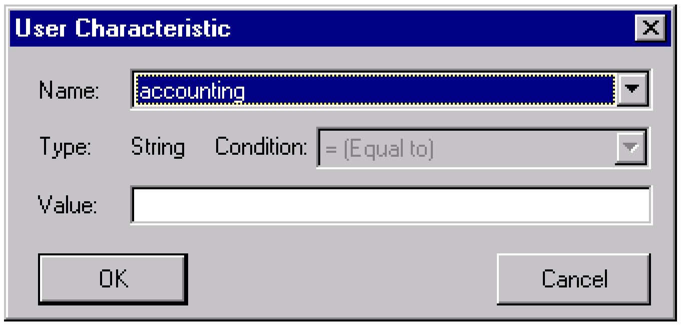
Table 7 - Machine Characteristics

Condition: This field allows you to select a comparison operator to be used when determining whether an Execution Queue meets the machine characteristic requirement. The following operators are available: Equal, Less, LessEqual, Greater and GreaterEqual. If the value is a string you may only use the Equal operator.

Value: This field allows you to enter a value that will be used in the above comparison. The data entered must meet the numeric or string value requirements of the machine characteristic chosen. For example, processor speed is numeric only. For string comparison purposes any data entered is matched from left to right until the data given is matched. This means you can enter partial data (for example: "Windows" would match both "Windows NT" and "Windows 2000").

Note: Machine Characteristic values vary quite a bit between operating systems and hardware platforms. Please check the *Release Notes* that accompanies each non-Microsoft Execution Agent for more specific information.

6.5.3.13 Job – Select User Characteristics


User Characteristic

Name: accounting

Type: String Condition: = (Equal to)

Value:

Figure 94 - Select User Characteristics

This dialog appears when you want to add or edit a user characteristic to a Job. User characteristics contain free-form data that you determine is important to your Job. User characteristics, like machine characteristics, are used to determine the appropriate Execution Queue. The example above allows the job author to select a machine queue where the characteristic “accounting” may be true or false.

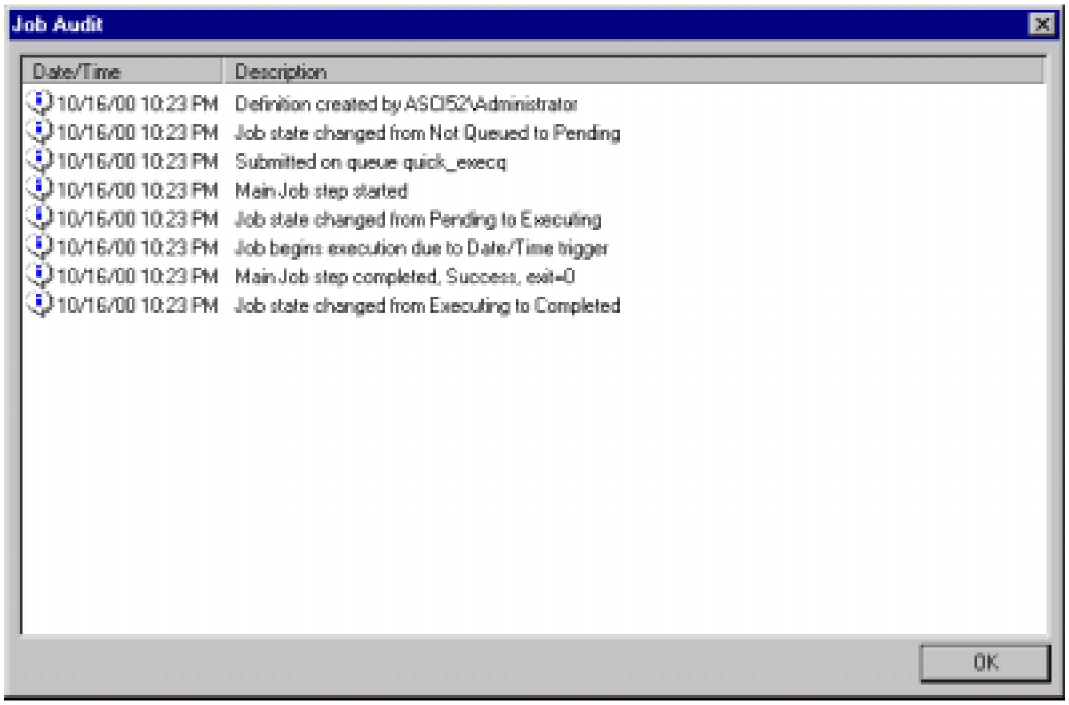
Name: This field contains a dropdown list of the user characteristics associated with your Execution Queues (through the Generic Queue).

Condition: This field allows you to select a comparison operator to be used when determining whether an Execution Queue meets the user characteristic requirement. The following operators are available: Equal, Less, LessEqual, Greater and GreaterEqual.

Value: This field allows you to enter a value that will be used in the above comparison. The data entered must meet the numeric or string value requirements of the user characteristic chosen. For string comparison purposes any data entered is matched from left to right until the data given is matched. This means you can enter partial data (for example: “Windows” would match both “Windows NT” and “Windows 2000”).

6.5.3.14 Job – Audit Trail

This dialog appears when you click the *View Audit Trail* button in the Job History tab. The audit trail is created on a per-job basis and contains all modifications and other ActiveBatch interactions. Each audit is contained in a single line in date and time sequence. An icon appears at the beginning of each audit to help visually signal the severity of the audit. The audit trail is read only and cannot be changed by the user.



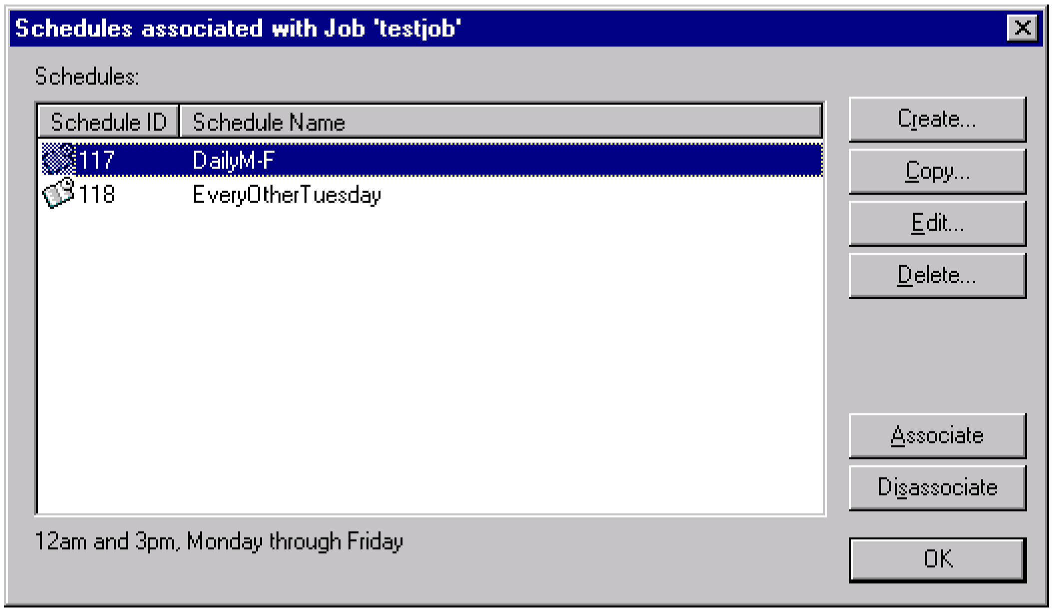
Date/Time	Description
10/16/00 10:23 PM	Definition created by ASCI52\Administrator
10/16/00 10:23 PM	Job state changed from Not Queued to Pending
10/16/00 10:23 PM	Submitted on queue quick_execq
10/16/00 10:23 PM	Main Job step started
10/16/00 10:23 PM	Job state changed from Pending to Executing
10/16/00 10:23 PM	Job begins execution due to Date/Time trigger
10/16/00 10:23 PM	Main Job step completed, Success, exit=0
10/16/00 10:23 PM	Job state changed from Executing to Completed

Figure 95 - Job Audits

6.5.4 Job Schedules

Job Schedules form the heart of the date/time trigger. ActiveBatch supports very flexible date/time scheduling. You can schedule both pattern (e.g. every 2 hours) and non-pattern (e.g. 1:31, 2:19) time periods. Schedules can be public or private and schedules, like other ActiveBatch objects, are secureable.

Schedules are typically created as a result of submitting a Job, however, because Schedules can be used by multiple jobs, you can manage Schedules at the Job Scheduler level as well.



Schedule ID	Schedule Name
117	DailyM-F
118	EveryOtherTuesday

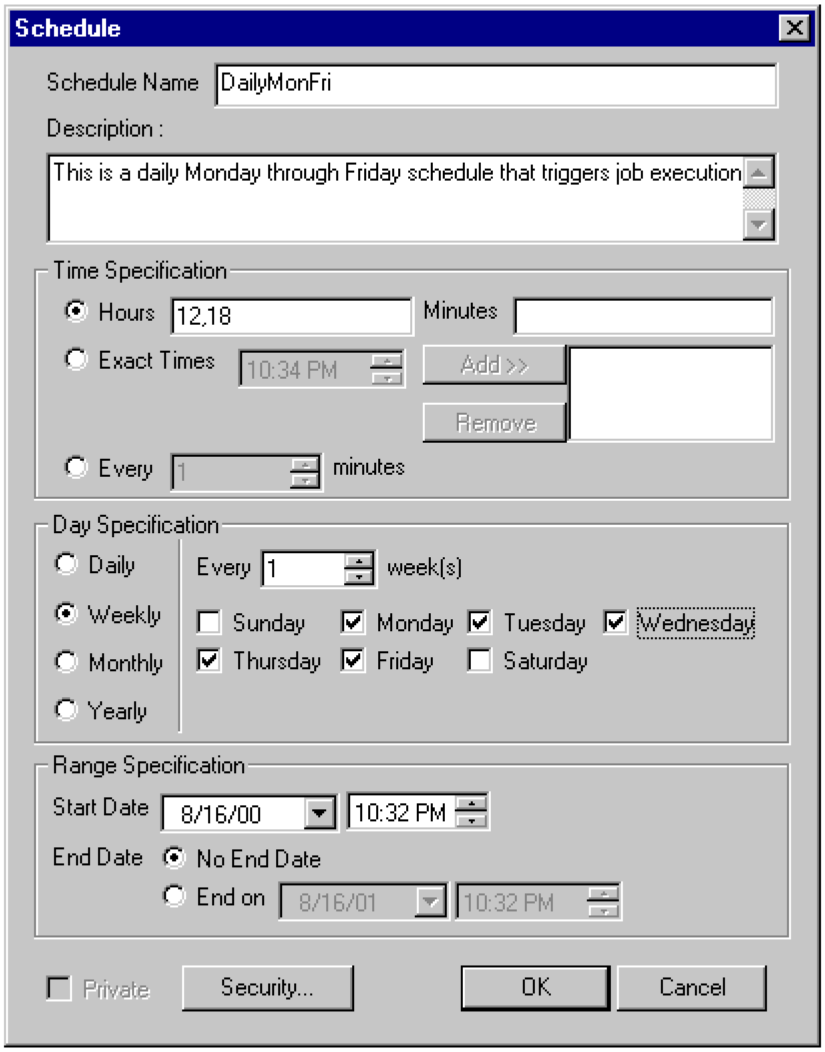
12am and 3pm, Monday through Friday

Figure 96 - Schedule List Dialog

The figure above lists all schedules that are associated with a Job. You will notice several action buttons on the right side. *Create* allows you to create a brand new Schedule. *Copy…* is similar to Create except that the newly created Schedule will use a selected Schedule as a template. *Edit…* allows you to edit an existing Schedule and make changes. *Delete* allows you to delete a Schedule off the system (don't confuse with Disassociate). *Associate* let's you associate a schedule with a job from a list of public schedules. *Disassociate* let's you disassociate a selected schedule. The last two action buttons won't appear if you manage schedules from the Job Scheduler object.

6.5.4.1 Create a Schedule

Creating a schedule is straightforward in ActiveBatch. You just need to think about what kind of date/time triggers you need for your job(s). When you have completed a Schedule simply click *OK* to confirm and apply the schedule or *Cancel* to cancel any changes made. The Schedule is also implicitly associated with the job if the Schedule creation was performed from a Job object. The *Security* button can also be used to add, edit or delete security access permissions.


Schedule

Schedule Name: DailyMonFri

Description: This is a daily Monday through Friday schedule that triggers job execution

Time Specification
[✓ Hours] 12,18 Minutes
Exact Times 10:34 PM
Every 1 minutes

Day Specification
Daily [✓ Weekly] Monthly Yearly
Every 1 week(s)
Sunday [✓ Monday] [✓ Tuesday] [✓ Wednesday] [✓ Thursday] [✓ Friday] Saturday

Range Specification
Start Date 8/16/00 10:32 PM
End Date [✓ No End Date] End on 8/16/01 10:32 PM

Private

Figure 97 - Create a Schedule Dialog

The figure above allows you to define a schedule. Schedules can be public or private. Schedules are also secureable objects. A Job can have one or more schedules associated as date/time triggers, so don't think you need to cram every possible date and/or time pattern into one schedule.

Schedule Name: This mandatory field represents the unique name of the Schedule.

Description: This optional field allows you to fully describe the schedule you're going to enter.

Time Specification: This section describes the *times* the job is expected to begin execution, not the dates. Dates are described in the next section, *Day Specification*.

The next three fields are mutually exclusive (radio buttons).

Hours: Enabling the radio button means you want hours and minutes level time specifications. The Hours and Minutes fields are combined and merged. So: Hours=0-23, Minutes=15 means a job is to run at :15 minutes past each hour. Hours=0,6, Minutes=15,45 means a job is to run at 00:15, 00:45 and 06:15 and 06:45.

Hours: This field represents hours in the day. The legal range is 0 through 23 (0=midnight). You may enter values as either a number or a comma separated list and/or a range value (i.e. 5-6). For example, "0,4,6-8" means midnight, 4am, 6 through 8am inclusive. By default, the assumed hours are 0-23 (every hour).

Minutes: This field represents minutes in the hour. The legal range is 0 through 59. As with hours, you may enter minutes as either a number, a comma separated list and/or a range value (i.e. 0-15). By default, 0 is assumed.

Exact Time: Enabling the radio button means you want to enter exact times. Typically you enter these times because they don't fit a pattern. For example, a job needs to run at 3:14 and 9:52. To enter exact times simply enter a time value in the time control box and then click the *Add*>> button to add the time to a list. Likewise to remove an already entered time, simply select the time from the scrollable list and click the *Remove* button.

Every: Enabling the radio button means you want to enter a interval time period specified by minutes and bounded by the minute. For example, 10 means a job is to execute every 10 minutes on a nn:10 minute boundary. For example, with a starting hour of 5 the possible execution times would be 5:10, 5:20, 5:30, etc. The legal range is 1-59.

Day Specification: This section deals with the day or date the job is to execute. Four mutually exclusive choices are available: Daily, Weekly, Monthly and Yearly. When a selection as described below is made, the right hand portion of the window changes to provide context specific fields.

Daily: Enabling the radio button means you want to schedule a Job on some type of daily basis. This field allows you to enter a number of days which will be interpreted as "every *n* days". For example, 1 means every day, 2 means every other day, etc.

Weekly: Enabling the radio button means you want to schedule a Job on some type of weekly basis. A field and seven (7) checkboxes will appear. The field indicates the frequency as in every *n* week(s). So 1 means every week, 2 every other week, etc. Clicking on a checkbox means you want the job to run on that day.

Monthly: Enabling the radio button means you want to schedule a Job on some type of monthly basis. Two (2) mutually exclusive fields will appear with a third frequency field.

Day: This field represents a day in the month. A legal range is 1-31 (note that for months ending in less than 31 days, a day greater than the number of days in the month simply means the last day in the month).

The… relative… day: This field contains two controls. The first is a relative schedule whose values are: First, Second, Third, Fourth and Last. The second is *days* value as in: Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Day, Weekday and Weekend. So you can combine the 2 fields such as Second Monday in the month or First Day in the month.

Of every… month(s): This field represents the frequency of the above entered monthly specification. Legal values are 1-12. So 1 means every month, 2 means every other month, etc.

Yearly: Enabling the radio button means you want to schedule a Job on some type of yearly basis. Two (2) mutually exclusive fields will appear.

Every: These two (2) controls represent a month and day specification.

The…relative…day…of every month: This field contains three controls. The first is a relative schedule whose values are: First, Second, Third, Fourth and Last. The second is a day value as in Sunday through Saturday and Day, Weekday and Weekend. The third is a frequency expressed in a month (January through December). So: The third Saturday in January or the Last Weekday in December is easily achievable.

Range Specification: This section indicates the effective date and time range this Schedule should be considered active.

Start Date: By default the starting date is today's date and time. You can however specify a different starting date and time.

End Date: You can either indicate that the Schedule is not to have an end date or you can specify an exact ending date and time. Note that even though a schedule may no longer be used for scheduling based on the effective date/time range, the schedule is not removed from the system and can always be modified to become effective once again.

Private: Clicking on this checkbox means the schedule is private and cannot be shared with any other job. Note: Jobs which use the /EVERY qualifier implicitly create private schedules.

Security… button allows you to assign security access permissions to the Schedule object. See Section 6.5.4.7 for more information.

6.5.4.2 Copying a Schedule

To copy a Schedule and make changes to the newly created object, simply select the Schedule and press the *Copy* button. You must have "Read" access permission to the Schedule object.

6.5.4.3 Editing a Schedule

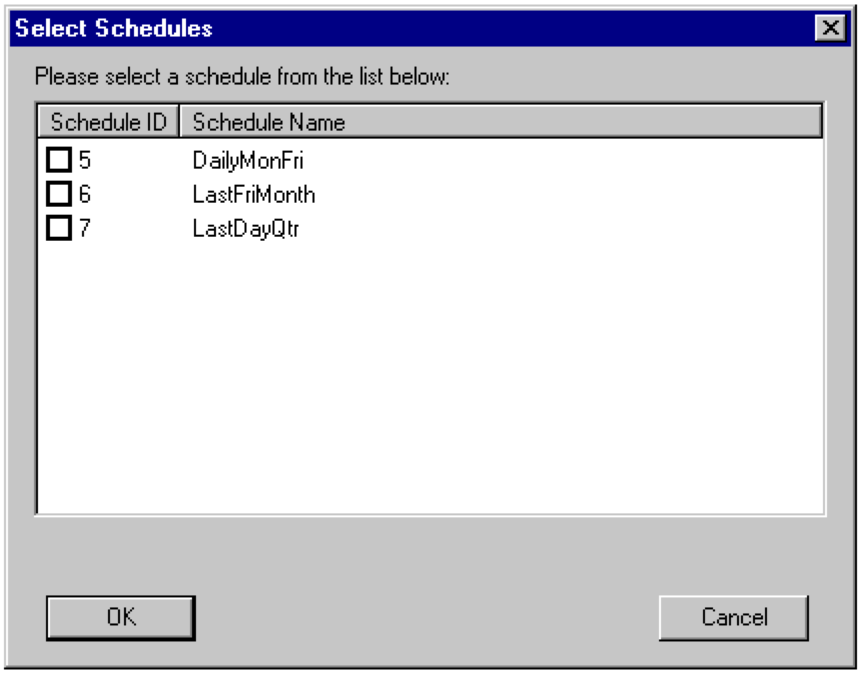
To edit an existing Schedule, simply select the Schedule and press the *Edit* button. You must have "Write" access permission to the Schedule object.

6.5.4.4 Deleting a Schedule

To delete an existing Schedule, simply select the Schedule and press the *Delete* button. A confirmation message box will appear asking you to confirm the deletion. You must have "Delete" access permission to the Schedule object. Note: Deleting a Schedule will automatically disassociate the Schedule from all jobs which are referencing it.

6.5.4.5 Associating a Schedule

To associate an existing Schedule with a job, press the *Associate* button.



Schedule ID	Schedule Name
☐ 5	DailyMonFri
☐ 6	LastFriMonth
☐ 7	LastDayQtr

Figure 98 - Select a Schedule Dialog

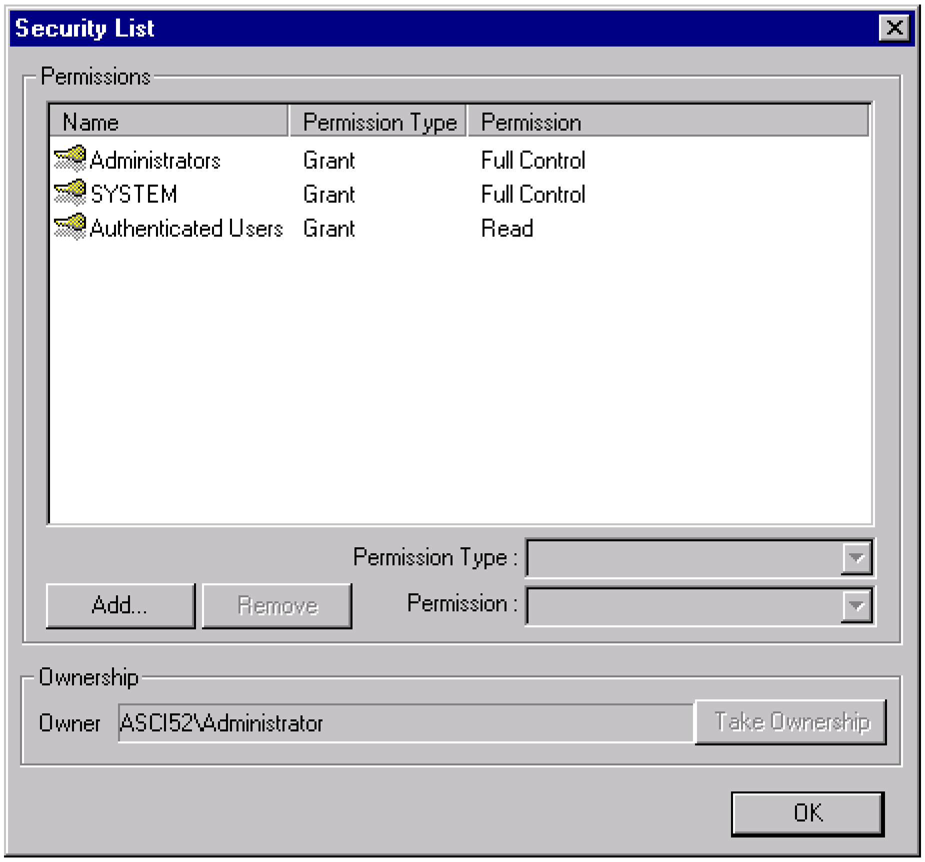
The above dialog box is displayed along with a scrollable list of all known public schedules. Simply select a Schedule and click the *OK* button. The Schedule will be added to the job. You must have “Read” access permission to both the Schedule and Job objects.

6.5.4.6 Disassociating a Schedule

To disassociate an existing Schedule from a job, select the Schedule and press the *Disassociate* button.

6.5.4.7 Securing a Schedule/Business Calendar

To apply access permissions to a Schedule or Business Calendar click on the respective *Security…* button on the dialog box.



Name	Permission Type	Permission
Administrators	Grant	Full Control
SYSTEM	Grant	Full Control
Authenticated Users	Grant	Read

Owner ASCI52\Administrator

Figure 99 - Schedule/Business Calendar Security Dialog

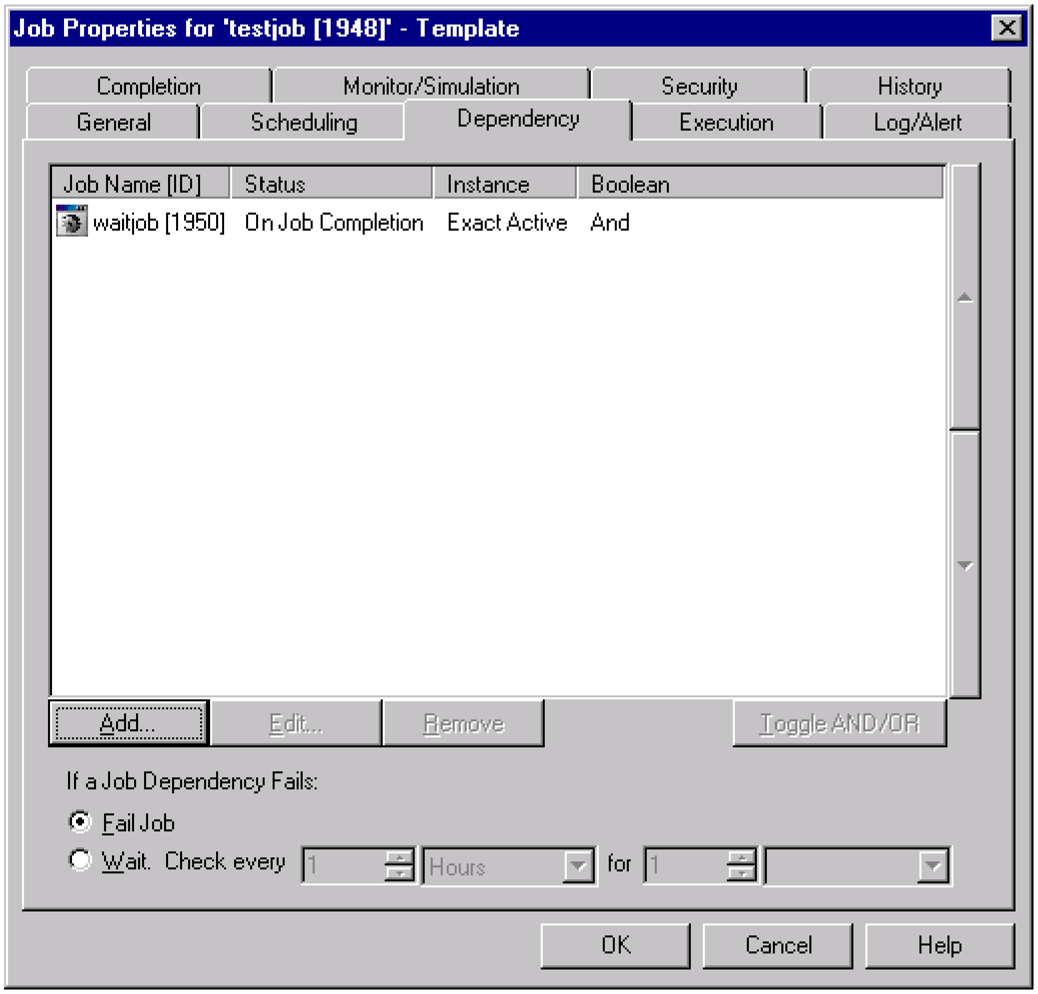
As with other object security the dialog above lists the existing permissions of any accounts that have specified access to the Schedule or Business Calendar. The following table lists the access rights and their descriptions.

Access	Description
Read	User is allowed to read the object.
Write	User is allowed to write the object.
Delete	User is allowed to delete the object.
Executive	User is allowed to update the object's security as well as take ownership.
Full Control	All of the above access is enabled.
No Access	User may not access the object.

Table 8 - Schedule/Business Calendar Access

6.5.5 Job Dependencies

A Job dependency is a relationship where one job must complete (either successfully or at all) before another job can begin execution. Section 6.5.3.4 discussed job dependencies when creating a new job. This section will expand on the Job Dependency dialog mentioned in that section.


Job Properties for 'testjob [1948]' - Template

Job Name [ID]	Status	Instance	Boolean
waitjob [1950]	On Job Completion	Exact Active	And

If a Job Dependency Fails:

[✓ Fail Job]

Wait. Check every 1 Hours for 1

Figure 100 - Job Dependency Tab Dialog

Figure 100 allows you to add, edit or remove a dependency. The dependencies listed are in the order they will be checked. The button *AND/OR* allows you to link dependencies such that "1st dependency OR 2nd dependency" establishes a logical ordering. The AND operator concatenates dependencies, for example, "1st OR 2nd AND 3rd" means that "the 1st job OR (2nd AND 3rd)". The *AND/OR* button applies to the selected job dependency. To change the order of dependencies use the UP and DOWN arrow buttons on the selected dependency.

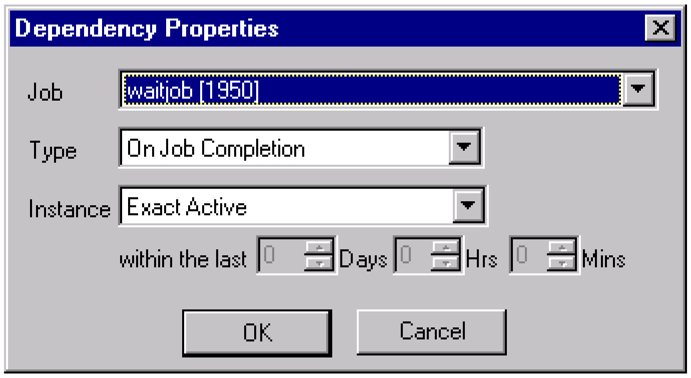
If a Job Dependency Fails: It is possible that the above conditions have not been met. For example, if Job "UpdateMaster" hasn't completed yet that would result in a "dependency failure". When a dependency fails you can indicate further processing.

Fail Job: This is pretty self-explanatory. ActiveBatch fails the job.

Wait: This indicates that we should wait. How long is determined by the next set of paired controls. **Check every** *number units* **for** *number interval*. *Units* is one of the following: Hours, Minutes or Seconds (the legal range of the number depends on the unit specified). *Interval* is one of the following: *Days*, *Hours*, *Minutes*, *Seconds*, *Times* or *Forever*.

6.5.5.1 Adding a Job Dependency

To add a Job Dependency, click the *ADD* button. The following dialog appears:



Job: waitjob [1950]
Type: On Job Completion
Instance: Exact Active
within the last 0 Days 0 Hrs 0 Mins

Figure 101 - Adding a Job Dependency Dialog

The information requested, in this dialog box, is to determine the job we're to be dependent on and what the dependency should be.

Job: This mandatory field contains a dropdown box listing all known active jobs, by job name, in the selected ActiveBatch Job Scheduler system.

Type: This mandatory field indicates whether the specified job must complete successfully or just complete.

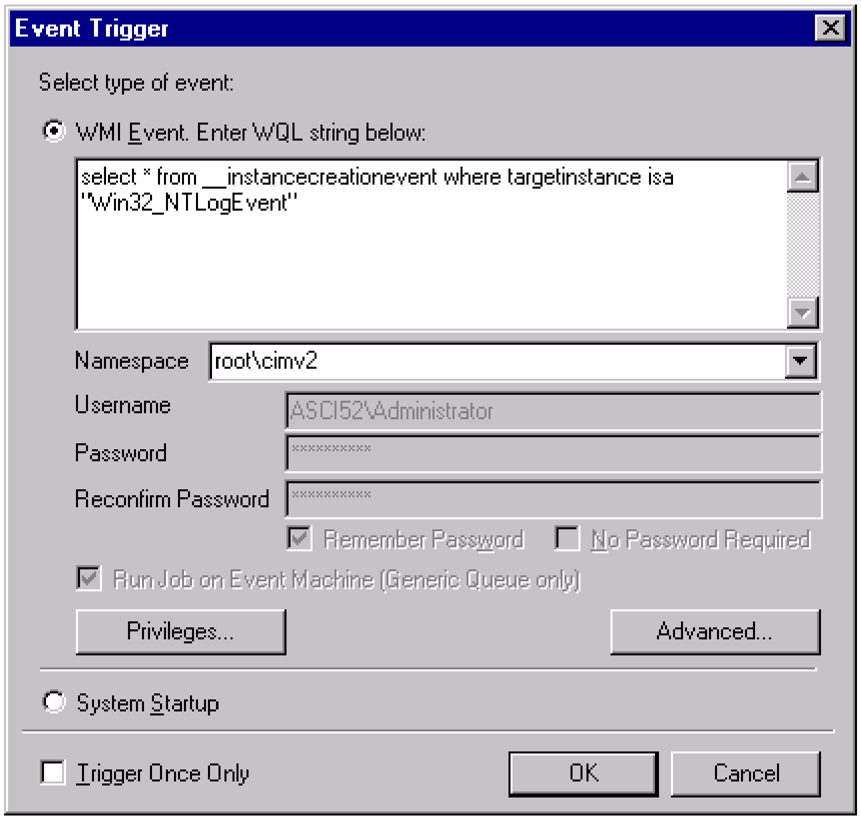
Instance: This field indicates how current the specified job instance must be to consider the job meeting the *Type* property. Possible choices are: *Exact Active*, *Last Instance* or *All Instances*.

> **Exact Active** means either the currently active instance or the last scheduled instance. This is the most precise setting.
>
> **Last Completed** means the last completed instance as specified by a user provided time period. When this dropdown is selected, the time control labeled "within the last" becomes active and you can set the days and hours/minutes as a time period for ActiveBatch to determine whether a completed instance meets these requirements.
>
> **All Instances** includes the preceding settings and expands the scope to include any instance of the job that completed. This is the most flexible setting.

Click *OK* to confirm and add the dependency to the list. Click *Cancel* to cancel the operation.

6.5.6 Event Triggers

ActiveBatch supports the integrated use of Microsoft Windows Management Instrumentation or WMI. WMI provides a common collection point for a diverse set of sources. WMI is Microsoft's implementation of WBEM. ActiveBatch is both an Event Provider and Event Consumer. First, the ActiveBatch Job Scheduler publishes a list of ActiveBatch events that it provides to WMI. ActiveBatch is also an Event Consumer. This means that ActiveBatch can register for any interested events and is notified by WMI when they occur. This section discusses the consumer aspects of ActiveBatch.


Event Trigger

Select type of event:

[✓ WMI Event. Enter WQL string below:]

```
select * from __instancecreationevent where targetinstance isa
"Win32_NTLogEvent"
```

Namespace: root\cimv2

Username: ASCI52\Administrator

Password: ***********

Reconfirm Password: ***********

[✓ Remember Password] No Password Required

[✓ Run Job on Event Machine (Generic Queue only)]

System Startup

Trigger Once Only

Figure 102 - Event Trigger Dialog

Note: WMI must be installed on the Job Scheduler machine for you to issue WMI Event triggers.

ActiveBatch allows job authors to indicate the events that a job may be interested in and can trigger execution of the job when the event occurs. After completing the requested information, click *OK* to confirm and apply or click *Cancel* to cancel any addition or changes to the list of event(s).

The dialog box requests that a WMI query be entered. For maximum flexibility, ActiveBatch supports the use of WQL (WMI Query Language, similar in syntax to SQL). Two (2) types of events are supported: System Startup and WMI.

System Startup: Clicking this radio button indicates that when the Job Scheduler machine is started the job should now run.

WMI Event. Enter WQL string below: This mandatory field takes a valid WQL statement describing the event you're interested in. All WQL statements begin with SELECT. You are not restricted in what you can enter however, you should not specify polling intervals that would adversely impact ActiveBatch and/or system performance. The example above shows a WQL query requesting an event be triggered each time a new NT Event Log entry is created (written to the NT Event Log).

Namespace: You must indicate the namespace to connect to. The Namespace specification is: \\machine\namespace, for example, ROOT\CIMV2 is the namespace for the local machine and \\TEST\ROOT\CIMV2 is the namespace for the TEST machine.

Username/Password: For a local machine (Job Scheduler machine) your logged in authentication credentials are used and WMI does not support the specification of different authentication credentials for a "local" machine. For non-Job Scheduler machines you must specify authentication credentials that will be used for WMI Event processing. As with other portions of ActiveBatch, you can indicate that the username and password are to be saved.

Run Job on Event Machine (Generic Queue Only): You can indicate to ActiveBatch that the job queue to select is the machine that actually generated the event. For this feature to work properly the job must be queued to a Generic Queue that contains at least one Execution Queue for the possible event machine. If no valid Execution Queue can be found that matches the event machine the job will not be run.

Trigger Once Only: If enabled, this event is triggered once and then disabled for the life of the job.

Privileges: This button allows you to enable or disable privileges that the selected WMI provider will use to execute your query. The sample query requires the SeSecurityPrivilege right.

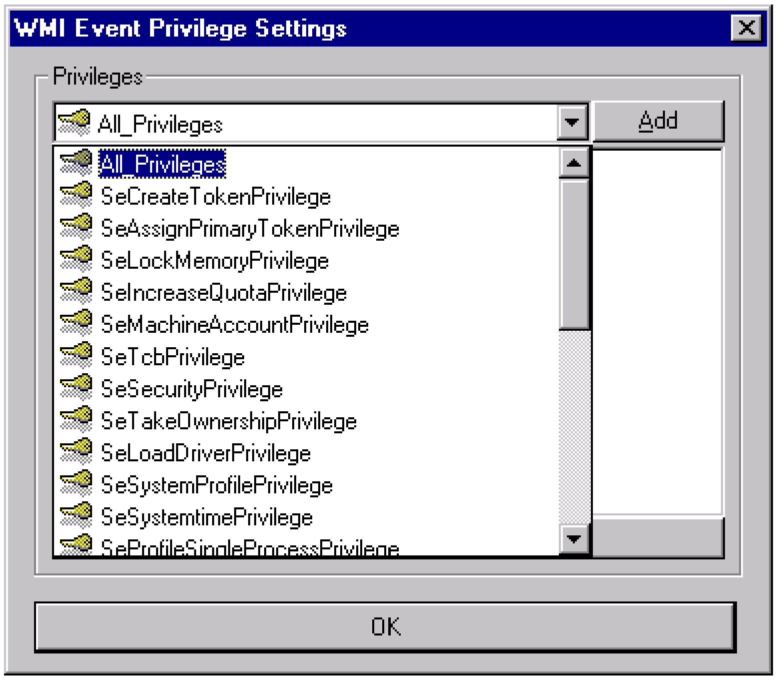
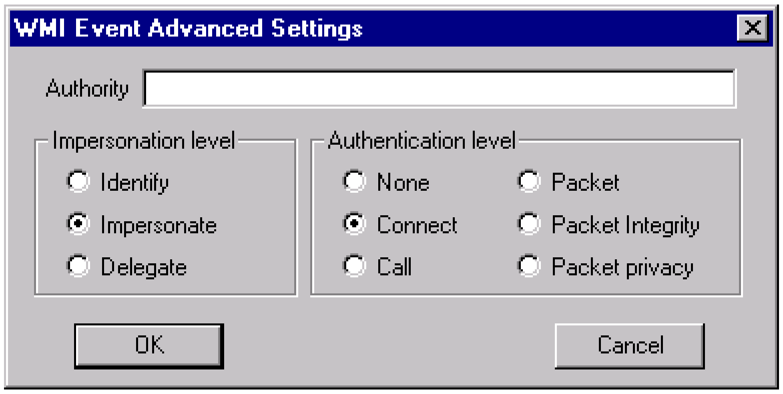


Figure 103 - WMI Event Privilege Settings

The above figure allows you to select one or more privileges (rights) that you would like to assert for the WMI Provider to properly execute the event query.

Advanced: This button allows you to select different authorization and security settings for the WMI query.



Authority:

Impersonation level: Identify [✓ Impersonate] Delegate

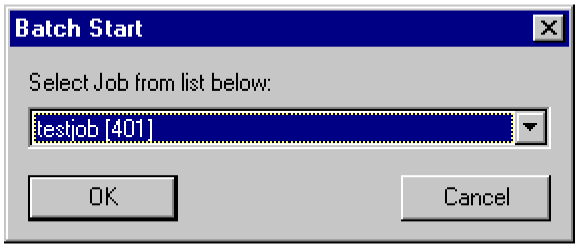
Authentication level: None [✓ Connect] Call Packet Packet Integrity Packet privacy

Figure 104 - Event Advanced Settings

The above figure allows you to change the Impersonation and/or Authentication levels. Advanced users can select different methods of securely using WMI.

6.5.7 Success/Failure Batch Start

ActiveBatch provides the ability to trigger execution of an existing batch job. You can determine which batch jobs to trigger based on how your job completes. This means you can trigger one set of jobs if your job completes successfully and another set if your job fails.



Select Job from list below:

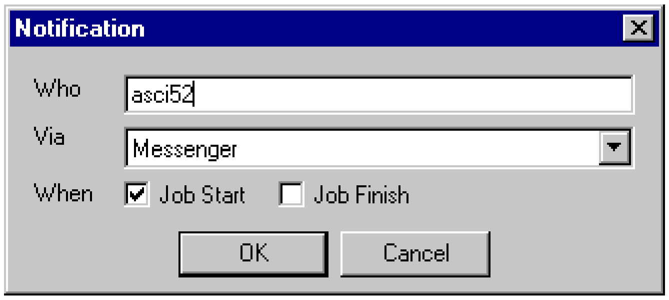
testjob [401]

Figure 105 - Batch Start Dialog

The dialog above is presented when you want to add a Batch Start trigger. A single dropdown list containing all eligible active jobs is presented. Only ActiveBatch Trigger enabled jobs are presented for selection. You need to select one and then click *OK* to confirm and apply the selection. Your selection is then added to the Batch Start list.

6.5.8 Job Notifications

When a job starts and/or completes you can specify a person (E-mail) or machine (NT Messenger) who should be notified.



Who: asci52
Via: Messenger
When: [✓ Job Start] Job Finish

Figure 106 - Job Notification Dialog

The dialog box above requests information about the user or machine that ActiveBatch should notify and how and when notification should be provided. Once the information is specified, click *OK* to confirm and apply the selection. Otherwise, click *Cancel* to cancel the notification operation. The example above indicates that machine 'asci52' is to be notified when the job begins execution.

Who: Depending on the "Via" this will be either an NT machine or an e-mail address (i.e. person@company.com).

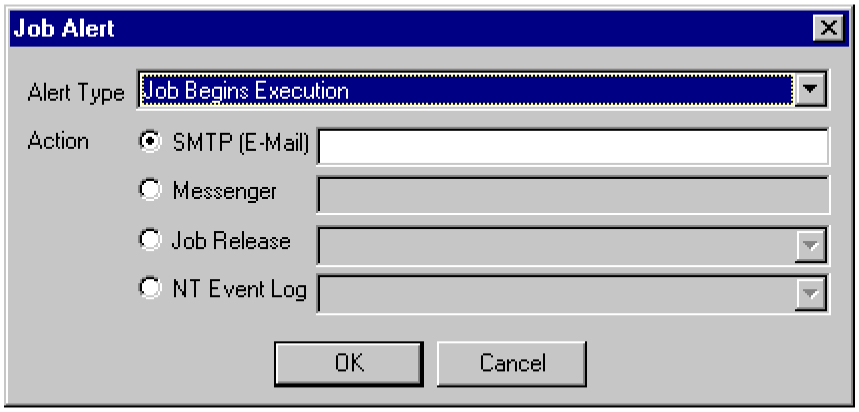
Via: This dropdown list contains two choices: E-mail (SMTP) or Messenger (the NT Messenger service must be active).

When: You can request notification for Job Start and/or Job Finish.

Note: You can also specify an NT username with the Messenger notification service, however, the Windows 2000/NT Messenger service will sometimes only send the message to the first user encountered when multiple users with the same name are logged in.

6.5.9 Job Alerts

Job Alerts, found on the Job – Log/Alerts dialog tab, is a facility for indicating what events constitute an alert which requires action. This is fundamentally different than notifications since a notification is just that, a message indicating something happened.



Alert Type: Job Begins Execution

Action: [✓ SMTP (E-Mail)] Messenger Job Release NT Event Log

Figure 107 - Job Alert Dialog

Enter the information requested and click *OK* to confirm and add the alert to a job. Click *Cancel* to cancel the operation.

Job Alert: This mandatory dropdown list contains all the alerts that can be specified for an ActiveBatch job. The following list is used:

Alert	Type	Description
Begin Execution	1	The job is beginning execution.
Completed	2	The job completed successfully.
Aborted	4	The job was aborted by manual intervention.
Completed Failure	3	The job completed in failure.
Pending Execution	7	The job is now pending execution.
Released	5	The job is released from a hold.
OnTrigger	6	The job has been triggered for execution.
RunTimeOverrun	8	The job has exceeded its expected elapsed time.

CPUTimeOverrun	9	The job has exceeded its expected CPU time.
RunTimeUnderrun	10	The job has underrun its expected elapsed time.
Restarted	12	The job has been restarted.
TriggerFailure	13	The job failed to start by trigger.
DependencyFailure	14	The job suffered a dependency failure.
Job Delayed	15	The job was delayed in starting on-time and outside a system policy acceptable delay.

Table 9 - Job Alerts

Action: One of four (4) mutually exclusive actions can be performed.

E-Mail (SMTP): An acceptable e-mail address. A special prefix may also be inserted for special formatting requirements. **BlackBerry** indicates the e-mail is being sent to a BlackBerry device running the ActiveBatch Wireless client. **Alphapager** indicates the e-mail is being set to a 150 character alphanumeric display pager. When a prefix is specified the format of the e-mail action is: prefix-keyword:e-mail-address (for example, blackberry:pager@company.com).

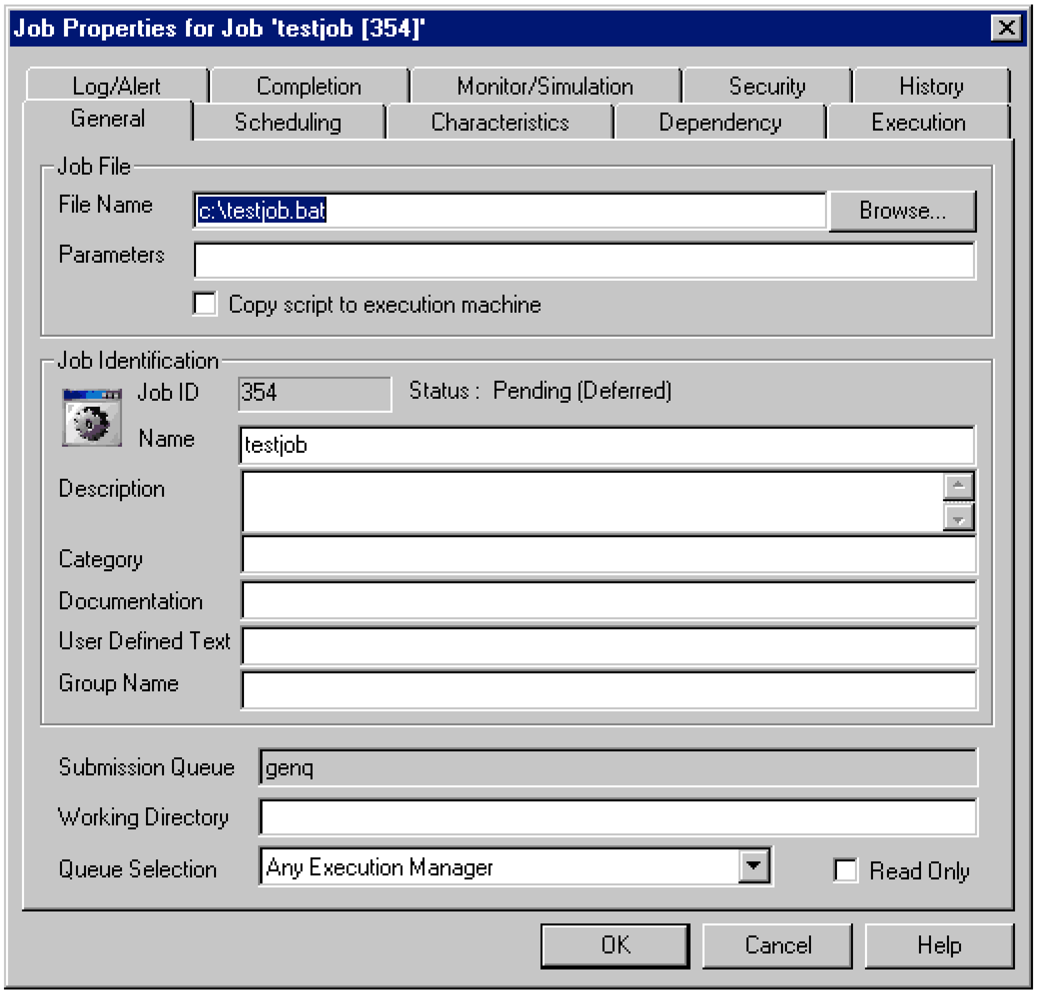
Messenger: An acceptable machine (or machine\username).

Job Trigger: This dropdown lists all known active jobs which can be triggered should this alert occur. Selecting one will cause a Trigger and Change Parameters operation to occur. Two formal parameters are passed to the job: JobID of the job that was the target of the alert and the alert number (see Table 9). Note: You must have "Trigger and Change Parameters" access permission and the job cannot reside on the OnDemand Queue. If the job already has parameters they will be overwritten for this job instance.

NT Event Log: This dropdown lists the acceptable NT Event Log machines. One of the following: Job Scheduler machine, Client and Execution Machine or you may enter a specific machine. See Section 3.3.1 for information concerning special Job Scheduler security requirements when selecting a machine other than the Job Scheduler to log NT Events to.

6.5.10 Modifying/Viewing a Job

To modify a Job you must hold “Write” access permission to the Job object. To perform the modification either double-click on the Job icon or right click on the selected Job icon and choose *Properties* or select the Job icon and choose *Jobs/Properties* from the menu bar. To avoid synchronization problems, ActiveBatch will hold the job during your modification and then release the job when you are done. This occurs, of course, only if the job is not already in a “hold” state.


Job Properties for Job 'testjob [354]'

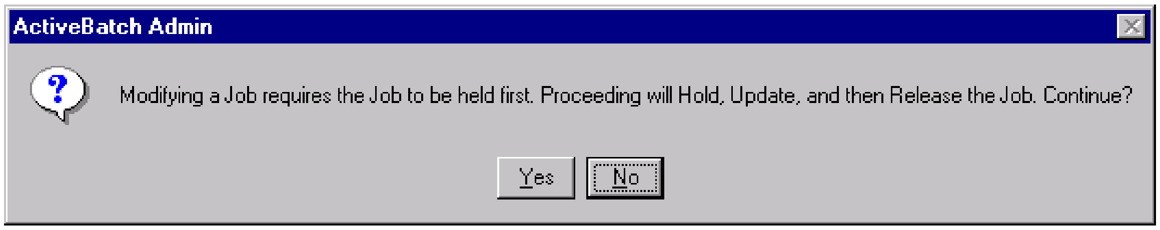
Job File
File Name: c:\testjob.bat
Parameters:
☐ Copy script to execution machine

Job Identification
Job ID: 354 Status : Pending (Deferred)
Name: testjob
Description:
Category:
Documentation:
User Defined Text:
Group Name:

Submission Queue: genq
Working Directory:
Queue Selection: Any Execution Manager ☐ Read Only

Figure 108 - Modifying a Job

You may always view a job regardless of the job's state. You can only modify a job when it is either in a Pending or Held state. If a Job is Pending and you modify the job you will need to confirm a temporary "Hold and Release" operation so that the data you modified can be applied to the job. ActiveBatch


ActiveBatch Admin

Modifying a Job requires the Job to be held first. Proceeding will Hold, Update, and then Release the Job. Continue?

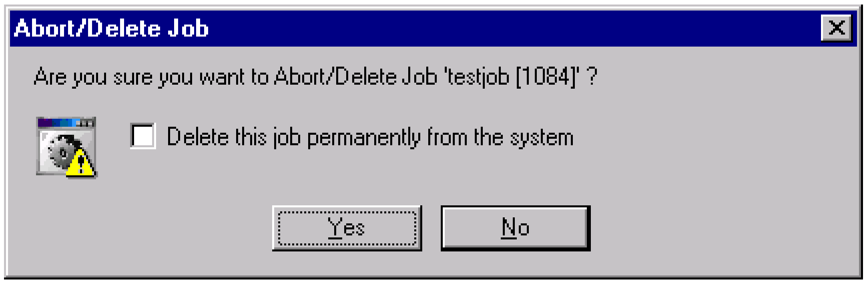
Figure 109 - Job Modification Confirmation

Clicking the *Yes* button confirms the operation and the job modification is attempted. Clicking the *No* button cancels the attempted modification. If the job is already held then the above confirmation does not occur since a special Hold/Release does not need to be performed.

6.5.11 Abort/Delete a Job

You may abort or delete a Job regardless of state. The difference between the two operations is that delete will also remove all job history information from the ActiveBatch database. Think of Abort as a "Soft Delete" retaining all job history information while Delete is a permanent irreversible action. The type of job that you want to abort or delete may also have far reaching effects. For example, if you abort a template you will be asked whether you also want to abort any active instances of the template. Likewise, if you delete a template you will be asked to confirm the implicit abort, and then delete, of all of the template's instances.

To abort or delete a Job you must hold "Delete" access permission on the Job object. To abort or delete a Job either right click on the selected Job icon and choose the *Abort/Delete* menu operation or select the Job icon and choose *Job/Delete* from the menu bar.

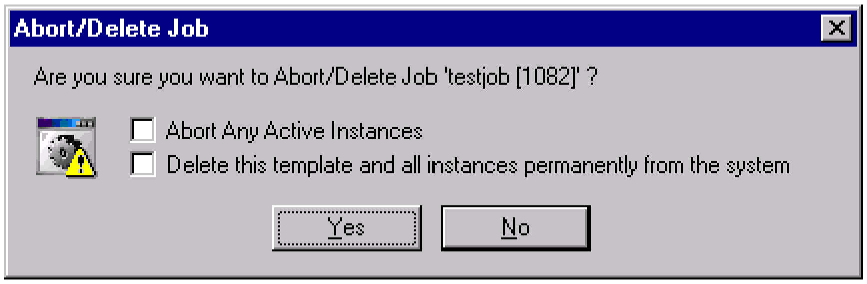

Abort/Delete Job

Are you sure you want to Abort/Delete Job 'testjob [1084]' ?

☐ Delete this job permanently from the system

Figure 110 – Abort/Delete a Job Dialog

When you abort or delete a Job, a confirmation dialog appears asking you to confirm the operation. Click *OK* to confirm or *Cancel* to cancel the operation.


Abort/Delete Job

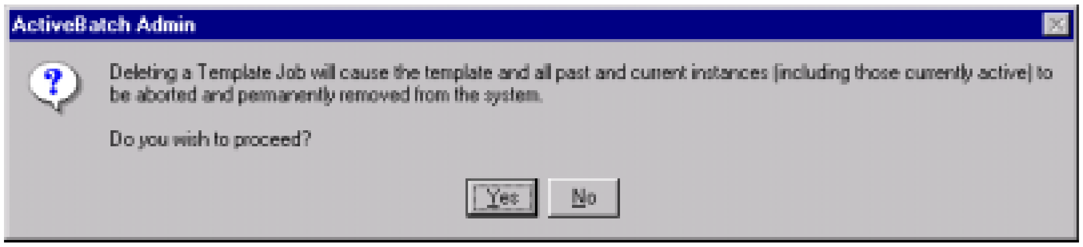
Are you sure you want to Abort/Delete Job 'testjob [1082]' ?

☐ Abort Any Active Instances

☐ Delete this template and all instances permanently from the system

Figure 111 -- Abort/Delete a Template Job Dialog

When you abort or delete a template job, you can further indicate whether you want to permanently delete the template and its instances or whether any active instances should be aborted. Checking *Delete this template…* will cause an additional confirmation to appear (Figure 112) since this is an irreversible action.


ActiveBatch Admin

Deleting a Template Job will cause the template and all past and current instances (including those currently active) to be aborted and permanently removed from the system.

Do you wish to proceed?

Figure 112 -- Delete Template Confirmation Dialog

6.5.12 Hold and Release a Job

Sometimes it is useful to hold a job and prevent its execution. ActiveBatch provides this operation through the Hold and Release facility.

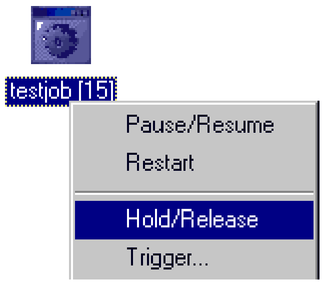


Figure 113 - Hold/Release a Job

Hold and Release are in-context operations. That means that ABATAdmin will choose the proper operation for you to perform based on the state of the job. If the job is not held, you will be able to hold it. If the job is held you will be able to release it. Hold and Release, while job oriented, do require “Manage” access permission for the Queue that the job is on as well as “Read” access permission on the selected Job object. To hold or release a job either right click on the selected Job icon and choose *Hold/Release* or select the Job icon and choose *Job/Hold/Release* from the menu bar. Note the job must be in a Pending state (so you cannot hold an executing job). Holding a template job causes no new instances to be created while the template is held.

6.5.13 Pause and Resume a Job

Where Hold and Release allow you to prevent/release a Job from/to executing, Pause and Resume are the counterpart commands you would use to suspend a job that is currently executing.

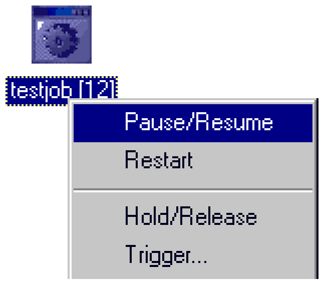


Figure 114 - Pause/Resume a Job

Like Hold and Release, Pause and Resume are in-context operations. You may only Pause an executing job. Pause and Resume require "Manage" access permission for the Queue the job is executing on as well as "Read" access permission on the selected Job object. To pause or resume a job either right click on the selected Job icon and choose *Pause/Resume* or select the Job icon and choose *Job/Pause/Resume* from the menu bar. Note you cannot pause a non-executing job.

Operating System Note: The implementation of Pause differs based on O/S platform. For Windows 2000/NT systems, only the primary thread of execution associated with the process is suspended. For multi-threading applications this may not have the desired effect. For non-Windows 2000/NT systems, the job process is suspended.

6.5.14 Restart Job Execution

This control function allows you to restart an already executing and active job from the beginning without the necessity for job resubmission. The restarted job begins in its entirety. This means that any “Pre” step is re-executed.

6.5.15 Resubmit Job

This function allows you to resubmit a job from the Runbook that has already completed and needs to be resubmitted for another run. Simply select the job to resubmit and select “Job\Resubmit”. A dialog box will appear (Figure 115) listing all the queues in all the Job Schedulers you are connected to. Select one queue and click OK. You will then have an opportunity to modify the job’s properties before the job is actually submitted for execution.

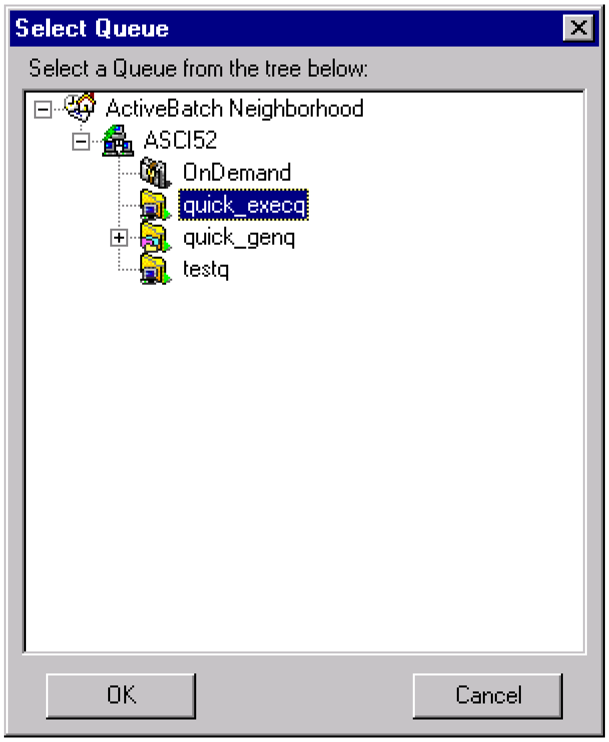
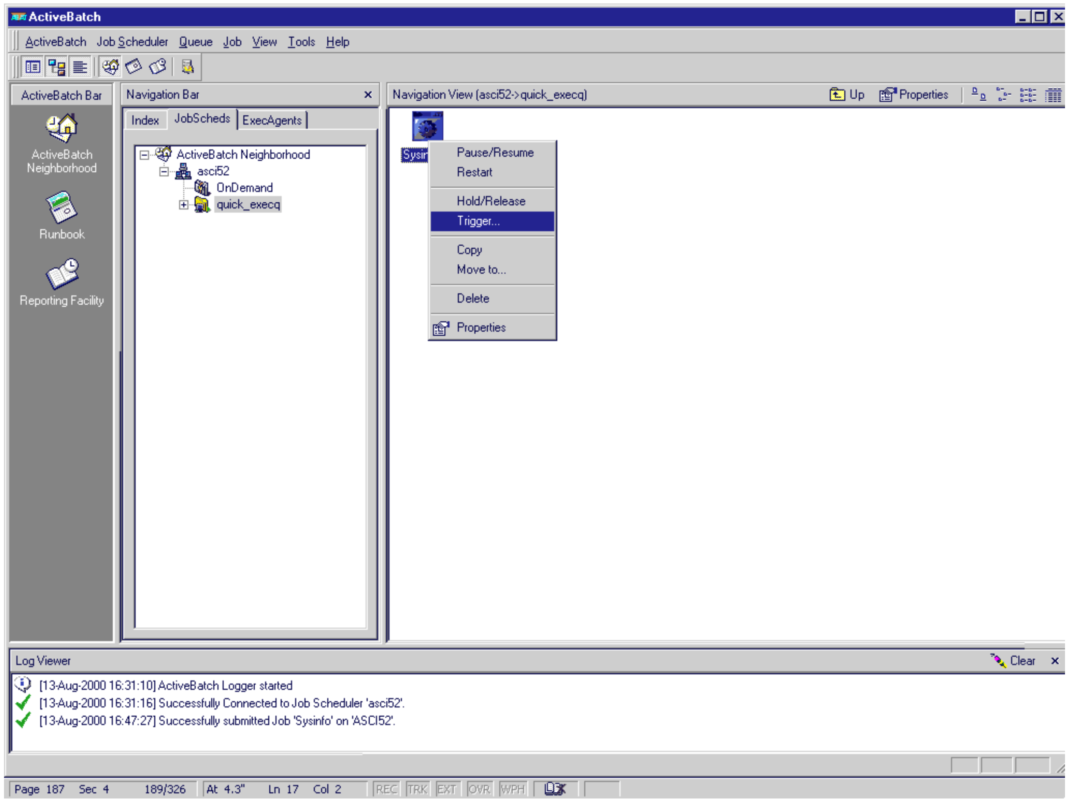


Figure 115 - Job Resubmit/Queue Selection

6.5.16 Trigger Job Execution

ActiveBatch allows the submission of jobs that can execute one or more times based on a variety of date/time or event triggers. The "Trigger" command allows a user or script to generate an ActiveBatch event that starts a job template executing. This trigger allows some interesting uses.



```
[13-Aug-2000 16:31:10] ActiveBatch Logger started
[13-Aug-2000 16:31:16] Successfully Connected to Job Scheduler 'asci52'.
[13-Aug-2000 16:47:27] Successfully submitted Job 'Sysinfo' on 'ASCI52'.
```

Figure 116 - Triggering a Job

To trigger a job either right click on the selected Job icon and choose *Trigger…* or select a Job icon and choose *Job/Trigger…* from the menu bar. Triggering a job requires you to hold at least "Trigger" access permission for the selected Job object. The job must have ActiveBatch Triggers enabled (see Job Scheduling tab). A job may be triggered from any ActiveBatch Queue however most trigger jobs are "On Demand" and reside on the OnDemand Queue (see Section 6.6 for more information on this special Queue).

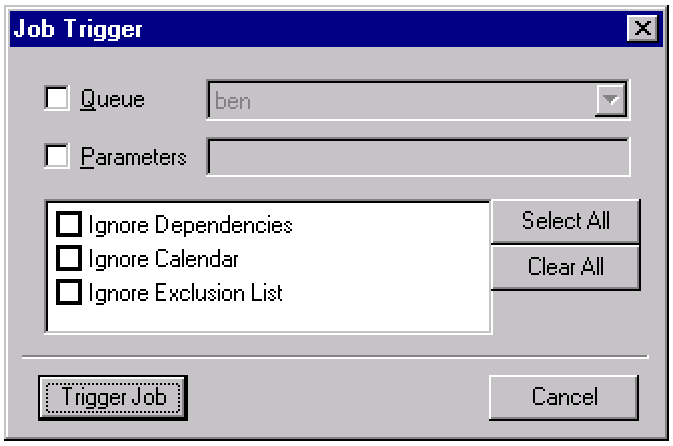


Figure 117 - Trigger Job Dialog

The dialog box above is presented to determine what actions should be taken with the trigger. If the Job resides on the OnDemand Queue you must specify a Generic or Execution Queue for the triggered job to execute on. Please note that the template job is not copied to the Queue you specified, only the instance of the job is created on the Queue. To trigger and specify a new Queue requires you to hold “Trigger and Queue” access permissions on the selected Job object. If the Job does not reside on the OnDemand Queue, you may trigger its execution in place (in other words, the job will start execution on the Queue it was submitted to).

Queue to trigger to: You may specify a Queue that is to receive an instance of this Job for execution. For a job that resides on the OnDemand queue you *must* specify a different queue. “Trigger and Queue” access permission for the selected Job is required.

Parameters: This optional field may be entered to provide new input parameters to the newly created instance of the triggered job. Note that the job template is not changed by this operation. You must hold “Trigger and Change Parameters” access permission for this Job to enter new parameters.

Ignore Dependencies: If selected, you are indicating that ActiveBatch is to avoid dependency checking and treat the selected job as though all dependency checks were successfully performed.

Ignore Calendar: If selected, ActiveBatch will ignore any Calendar restrictions and run the job.

Ignore Exclusion List: If selected, ActiveBatch will ignore any date/time scheduling exclusions.

6.5.17 Copy and Paste a Job

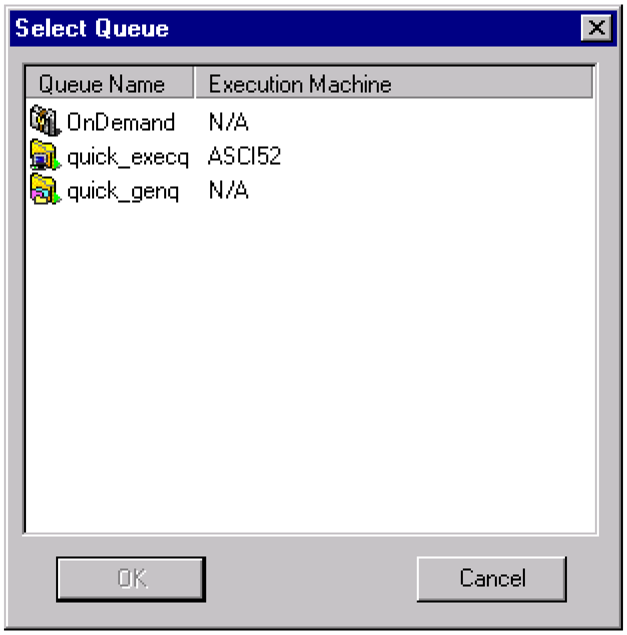
This pair of operations are actually performed by the ABATAdmin program as opposed to ActiveBatch itself. Sometimes you want to create a job using most of the information found in another job. ABATAdmin let's you do just that.

To copy a job either right click on the *selected* Job icon and choose *Copy* or select a Job icon and choose *Job/Copy* from the menu bar. This action makes a copy of the Job object.

To paste the copied job, select a Queue for the job you want to paste and then right click on the selected Queue and choose *Paste as New Job*. The New Job property tab dialog will be displayed allowing you to change any properties concerning your new job. All job properties will be set in the newly pasted job except that your username and security credentials (if any) will be used. If they are not present you will need to enter them. You will also become the owner of the newly created job. Clicking *OK* confirms the job submission. Clicking *Cancel* indicates that you want to "undo" the paste operation and not submit the job. The job will, however, remain in the copy buffer. Please note that you must have "Read" access permission to copy the initial Job object.

6.5.18 Moving a Job

Sometimes, for various operational reasons, you need to move a Job from one Queue to another. Moving a Job requires you to hold “Write” access to the selected Job object.



Queue Name	Execution Machine
OnDemand	N/A
quick_execq	ASCI52
quick_genq	N/A

Figure 118 - Moving a Job

To move or requeue a job either right click on the Job icon and select *Move* or select a Job icon and choose *Job/Move*. A list box appears listing all queues. Select one to perform the move operation.

6.6 OnDemand Queue

ActiveBatch provides a special queue named *OnDemand*. The OnDemand Queue is a special type of Execution Queue that is not associated with a machine. For example, let's say you wrote a script to start up an NT service for a machine. If you wanted to execute the job on a specific machine you would simply submit the job to that machine's Execution queue. But what if you didn't know what machine would need to use your script until just before the job is to run? That's the driving force behind the OnDemand queue. Jobs that are submitted to the OnDemand queue enter a special "wait" state and become template jobs. Remember that a template job doesn't actually execute. Instead the instance(s) of the template execute. Jobs that are on this queue can't create instances until a Trigger command is issued. The Trigger command provides the facility for starting an instance of a template and, of particular importance to OnDemand job templates, the ability to specify a queue for execution. This means you could submit the script that starts a service and then, later on, issue a Trigger command indicating what queue (machine) the job instance should execute on. Remember that only the instance moves to the newly specified queue not the template. This means that you could have the same script run on different queues (machines) at the same time, based "on demand".

It is important to remember that jobs never execute on the OnDemand queue. You must always trigger such jobs for execution to a different queue.

6.6.1 Security Implications

Since the ability to Trigger an OnDemand job is so powerful in scope, ActiveBatch provides several different access permissions for jobs which are "triggerable". First, an ActiveBatch job must indicate that it is willing to support the ActiveBatch Trigger command. This property can be found on the Job Scheduling tab of the ABATAdmin Job Dialogues. This property allows you to enable or disable the potential use of this command. Most users will want to enable this feature. Next, you will want to control who can Trigger a job and what Trigger features can be enabled. If you examine Table 6 you'll notice three (3) Trigger access permissions. Let's discuss each in more detail.

6.6.1.1 Trigger Access

This level of access simply means that a named user or group is allowed to start the job. No other aspects of the job can be changed. Trigger level access permission alone makes little sense if you are attempting to trigger an OnDemand job since a major requirement is the ability to specify a queue for the job to execute on.

6.6.1.2 Trigger with Queue Change Access

This level of access allows a user to both trigger and specify a queue that the job instance should execute on. This access permission is what is typically granted to a user who must trigger a template OnDemand job. The user can now both trigger and specify a queue (machine) for execution.

6.6.1.3 Trigger with Change Parameters Access

This level of access indicates that a user is allowed to trigger a template job and pass a new set of input parameters to the job instance. The input parameters could be anything that helps the job execute properly.

By allowing different access permissions for the Trigger command, a job author can precisely indicate what can be triggered and where the job can execute.

6.6.2 Second Party Job Release

Typically, the author of the job will issue a Trigger command, usually from within another batch job stream that's been written. This means the same user is issuing a Trigger from another job using the same security context. Since the Trigger command requires access permissions, a very nice benefit can be obtained from letting *other users* issue a Trigger command. Let's look at our earlier example on starting a Windows NT service. Normally starting a service would require Administrator level access and most administrators with good reason won't just add a user to that group even if, on occasion, there's justification in doing so. On the other hand, what if you (an administrator) could submit a job to the OnDemand queue and allow a normal user the ability to trigger your job for execution on his machine? That way you wouldn't have to give him your security credentials (a bad thing) and you wouldn't have to add him to any Admin groups. That is precisely what Triggering an OnDemand job allows. You simply submit a job with your security credentials to an OnDemand queue and allow (using ActiveBatch security) one or more users to issue a "Trigger with Queue Change" access.

6.7 Job Checkpoint and Restart

ActiveBatch provides several features to allow jobs to automatically restart or failover to an available machine. By default, jobs that don't have restart enabled are considered to have failed when the machine they execute on fails. Two types of job restart are provided (see Section 6.5.3.7). *Restart on Reboot* is the more typical and simply means that the job is to be restarted when the machine the job was executing on is rebooted and available. *Failover Restart* is applicable when the job is placed on a Generic Queue and can be re-dispatched to another available machine. Where "Restart on Reboot" means the job must wait until the execution machine is rebooted, "Failover Restart" means the job can be immediately re-dispatched. When a job is restarted, its "pre" step is executed, followed by the "Main" and "Post" steps. Please note that the ABAT_RESTART variable is set to YES to indicate the job is operating under a restart condition (see Table 5 - Environment Variables for Job Steps in Section 6.5.3.5 for more information).

ActiveBatch supports the concept of *Checkpointing*. A checkpoint represents non volatile data that is associated with the batch job and saved by ActiveBatch. The last checkpoint value is made available whenever the batch job is restarted. Of course, the actual data and timing of checkpoints are purely up to the batch job itself. To declare a checkpoint you invoke the AbatChkPt program. For most machines the syntax is:

```
ABATCHKPT "Checkpoint-data"
```

ABATCHKPT is the name of the checkpoint program and the data you want to checkpoint appears after the command (with an intervening space). The string specified does not have to be quoted unless the data contains embedded spaces. *This command can only be issued within an ActiveBatch job*.

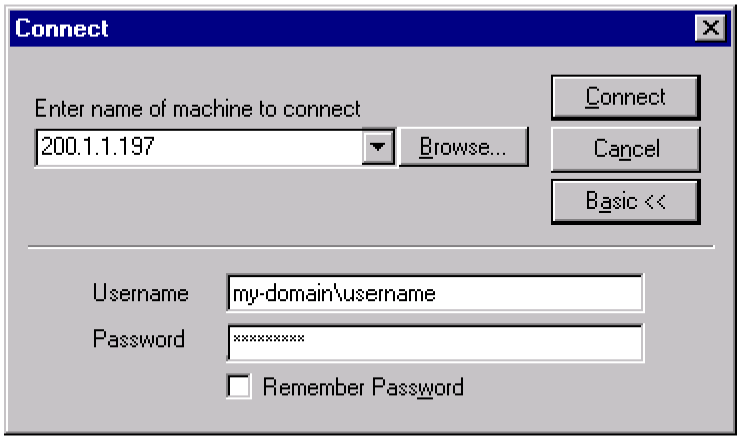
To retrieve the checkpoint value, you must check two ActiveBatch environment variables (or logical names), ABAT_RESTART and ABAT_CHECKPOINT. Both must be set to YES for the checkpoint value (ABAT_CHECKPOINTVAL) to be considered safe and available.

7 ActiveBatch Features and their Applications

This section more fully describes various ActiveBatch features and in particular provides context and examples for how you might employ them.

7.1 Advanced Connections

While most client to Job Scheduler connections will probably use your local credentials, don't forget that you can also specify different security credentials when necessary or desirable.



Enter name of machine to connect: 200.1.1.197

Username: my-domain\username

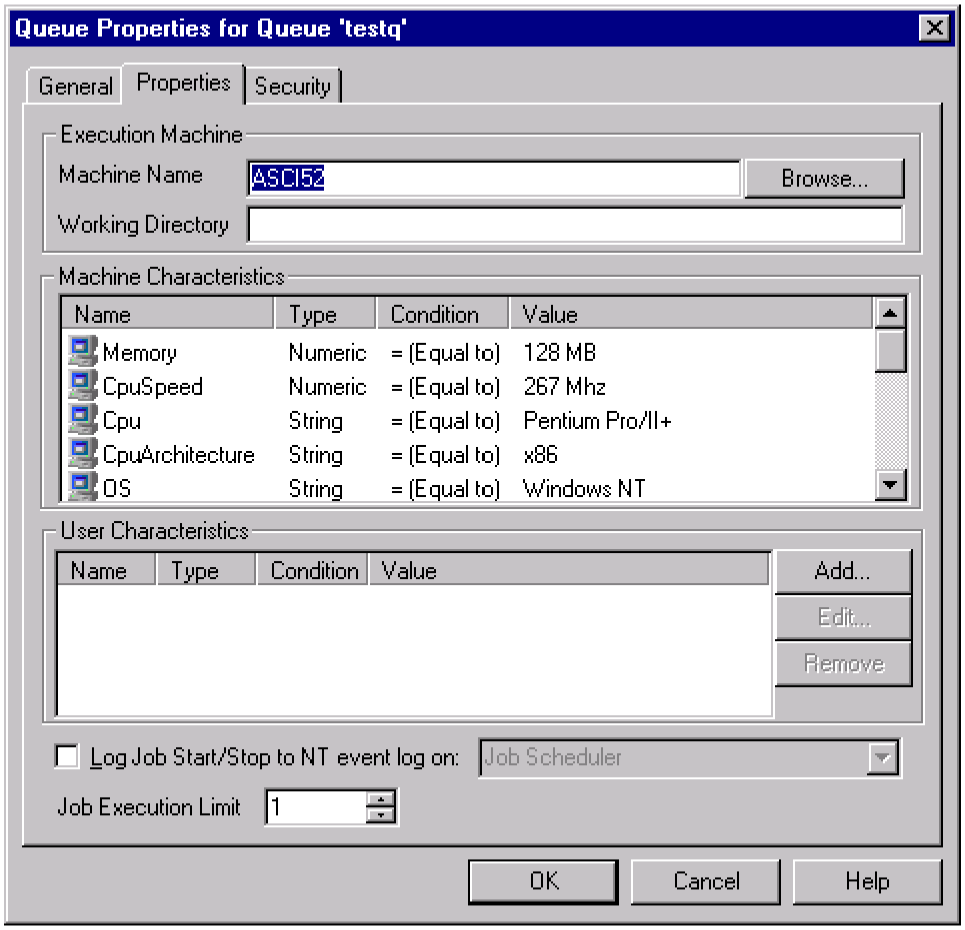
Password: **********

The dialog box shows an advanced connection using an IP address and a user-supplied username/password. If you made this connection a lot you could also check the "Remember Password" box. Advanced connections are useful when you're accessing machines over the Internet from home and you need to specify security credentials that certainly wouldn't be the same as those on your home computer. Another good situation would be accessing a Job Scheduler in another domain or simply specifying another user's security context.

7.2 Setting Up Your Queues

7.2.1 Execution Queues

The first thing you'll need to do to become familiar with ActiveBatch is to setup at least one Execution Queue. Ultimately all jobs need to find their way to an Execution Queue to actually execute. That's because an Execution Queue represents a machine. Figure 119 shows those properties that are specific to an Execution Queue.



Queue Properties for Queue 'testq'

Execution Machine

Machine Name: ASCI52

Working Directory:

Machine Characteristics

Name	Type	Condition	Value
Memory	Numeric	= (Equal to)	128 MB
CpuSpeed	Numeric	= (Equal to)	267 Mhz
Cpu	String	= (Equal to)	Pentium Pro/II+
CpuArchitecture	String	= (Equal to)	x86
OS	String	= (Equal to)	Windows NT

User Characteristics

Name	Type	Condition	Value

☐ Log Job Start/Stop to NT event log on: Job Scheduler

Job Execution Limit: 1

Figure 119 - Execution Queue Properties

While you will want at least one Execution Queue for each machine that you intend to execute jobs on, it is important to note that you can create multiple Execution Queues for a single machine. The critical properties for an Execution Queue are listed in this figure. The **Machine Name** as you'd imagine is most important. This property indicates what machine is to be associated with this named queue. It is important to note that ActiveBatch uses the client's machine name as the default when creating an Execution Queue. So if you want to associate a specific machine to an Execution Queue don't forget about changing that property! Next, is the **Job Execution Limit** property. This property controls the number of simultaneously executing jobs that this queue can support. With the value at one, once a job begins executing, all other jobs for that queue, will enter a "Pending (Queue Busy)" state. If you needed to ensure that any jobs submitted to a queue executed in a serial fashion you might create a queue with the Job Execution Limit set to one. Since you can have multiple Execution Queues associated with the same machine, you could have another queue with a higher execution limit.

Another interesting set of properties is **Machine** and **User Characteristics**. Machine characteristics refer to a fixed set of characteristics that describe some of the hardware and software properties of the execution machine. User characteristics allow you, the user, the ability to establish any property that you want. The benefit of these characteristics or properties is to schedule a job to run on the right machine and/or at the right time. Machine and User Characteristics are examined when an Execution Queue is associated with a Generic Queue. These characteristics are then used to pick the correct set of machines and ultimately the correct machine for which to run a job. We'll talk more about this in the next section on Generic Queues.

An Execution Queue must be started for jobs to execute. A queue in the "starting" state means that the execution machine is either not accessible or that the execution agent software has not been started. You should treat "starting" as "stopped" since operationally no jobs will run. The "stopped" state means that either a manual or timed stop operation was issued against the queue. Jobs are not started for execution when a queue is in the "starting" or "stopped" state.

7.2.2 Generic Queues

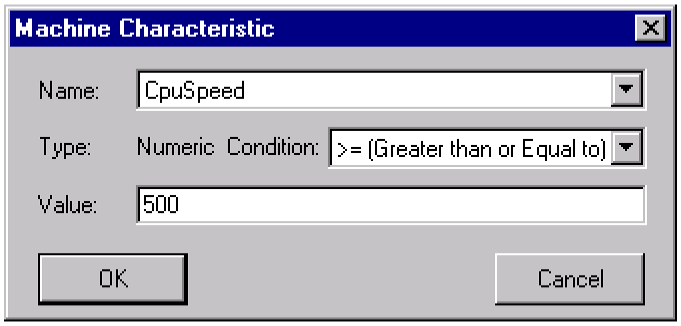
After you've tried using Execution Queues you'll want to use some of the advantages of Generic Queues. A Generic Queue is made up of one or more Execution Queues. Those queues and the machines they represent should be of the same type. In other words, NT machines would be categorized with other NT machines, Tru64 UNIX machines would be grouped with other Tru64 UNIX machines. So a Generic Queue is a type of virtual machine. Why is that so great? Essentially because a Generic Queue forms an abstraction layer between the queues and the real machines they represent and that layer can be greatly exploited. When a job is submitted to a Generic Queue you have two main choices during submission. You can indicate that one Execution Queue is to be selected out of any eligible queues. This is known as the *Any Execution Machine* selection method. You can also indicate that a job submitted to a Generic Queue be broadcast to all eligible queues. This is known as the *All Execution Machines* selection method. Since most users want a job to execute on one machine we'll discuss the "Any" selection method first.

ActiveBatch uses the following method for selecting an Execution Queue member within a Generic Queue ("Any" selection). First, the queue must be in a "Started" state. Next, the Job's Machine and User Characteristics (as described in the next section) must match. Lastly, the Queue Scheduling Algorithm is used to pick a winner.

For the "All" selection method only the Machine and/or User Characteristics must match since the Job will be broadcast to all eligible queues (and their machines).

7.2.2.1 Characteristics

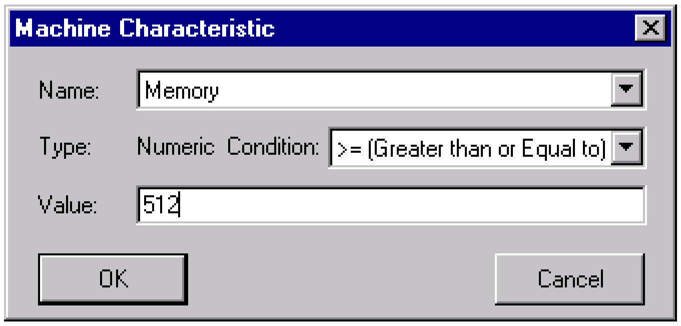
Machine and/or User Characteristics can be used to determine a set of eligible Execution Queues from which to execute a given job. For example, if you needed a Job to execute on a machine with at least 500mhz of CPU speed and 512 MB of memory, you would submit a job with the following machine characteristics:


Machine Characteristic

Name: CpuSpeed

Type: Numeric Condition: >= (Greater than or Equal to)

Value: 500

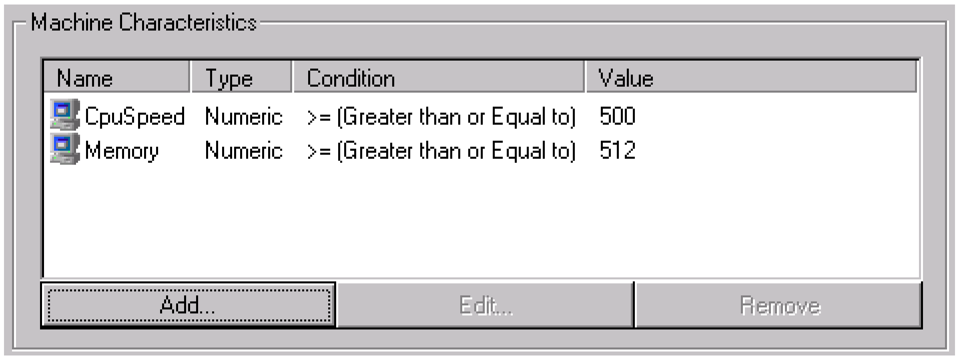

Machine Characteristic

Name: Memory

Type: Numeric Condition: >= (Greater than or Equal to)

Value: 512

The Job's Machine Characteristics section would look like this:



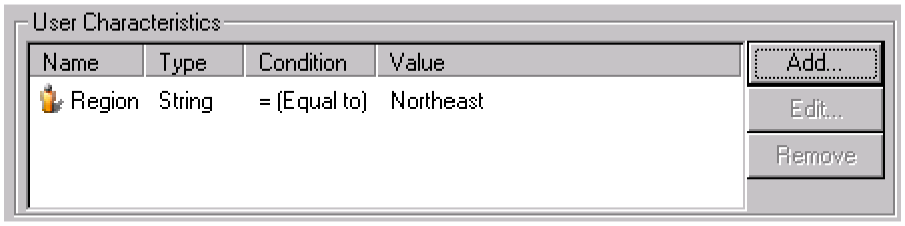
Machine Characteristics

Name	Type	Condition	Value
CpuSpeed	Numeric	>= (Greater than or Equal to)	500
Memory	Numeric	>= (Greater than or Equal to)	512

You could further refine the selection by Processor Architecture or Processor Type. For example you might want to run your job only on Pentium chips.

Not all Machine Characteristics deal with hardware. You could decide that only Windows NT V4 machines using at least "ServicePack 5" be considered eligible for running your job. As you can see you have quite a bit of leverage as it relates to selecting eligible Execution Queues (and the machines they represent).

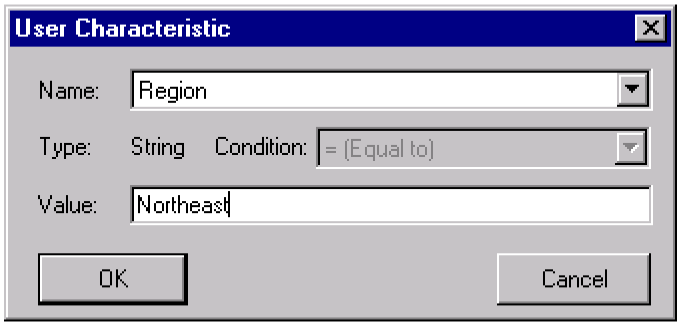
In addition to Machine Characteristics you can also create and then employ User Characteristics. A User Characteristic is a free-form property that you create on an Execution Queue and then use when you submit a Job. For example, let's suppose you wanted to define a "Region" property that associated an Execution Queue (and its machine) within an area of the country. That User Characteristic might look like this on the Execution Queue property tab:



User Characteristics

Name	Type	Condition	Value
Region	String	= (Equal to)	Northeast

If you had Execution Queues with different "Region" values (for example, Southeast, Northwest, etc). You could submit a job with the following User Characteristic request:



Name: Region

Type: String Condition: = (Equal to)

Value: Northeast

This would further limit the number of available systems to just those in the Northeast.

7.2.2.2 Queue Scheduling Algorithms

Once the set of eligible Execution Queues is determined the Queue Scheduling Algorithm takes over. Three (3) algorithms are available for scheduling a job to an Execution Queue: Sequential, Round-Robin and Workload Balancing.

Sequential means that the first Execution Queue encountered will be selected unless the execution job limit has been reached. In that case the next queue will be selected.

Round Robin means that the next Execution Queue after the last selected will be used.

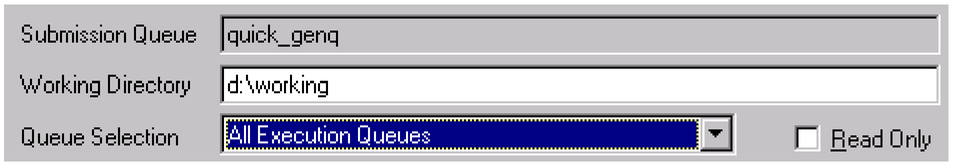
Workload Balancing means that the best Execution Queue as it relates to CPU, Memory and other resource utilization factors will be used.

We suspect that most customers will use the Workload Balancing and Sequential algorithms. Workload Balancing offers the advantage of ensuring best resource utilization among a number of machines. CPU speed and memory capabilities are also used in the algorithm such that unequal machines are used properly. In other words, 2 machines that have identical CPU and Memory resource utilization will not be treated the same if one of the machines has more memory capacity and/or raw CPU speed. Workload balancing helps to ensure machine scalability since you can always add additional Execution Queues as members to a Generic Queue without changing any aspects of the job. This is another advantage to abstracting the members of the Generic Queue. From a management point-of-view you can always add or remove Execution Queue members.

The Sequential algorithm offers a different "tack" on scheduling jobs to machines. With Sequential, the first execution machine will be used until it reaches the maximum job execution limit for the queue. Subsequent jobs are queued to the next available Execution Queue. If all queues are in use, the job will remain queued to the Generic Queue until an Execution Queue becomes available.

7.2.2.3 Job Broadcast

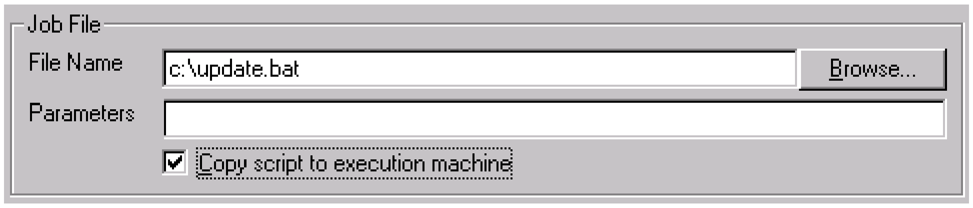
The previous sections dealt with the perspective of scheduling a job on _one_ Execution Queue member within a Generic Queue. This section deals with scheduling a job on more than one Execution Queue member(s) of a Generic Queue. This capability is known as the “All” queue selection method or Job Broadcast. Figure 120 shows a portion of the Job General property tab that enables this capability.



Submission Queue: quick_genq
Working Directory: d:\working
Queue Selection: All Execution Queues
☐ Read Only

Figure 120 - Job Broadcast

In the “All” position, all eligible Execution Queues will be used for job scheduling purposes. A good application of this feature might be to broadcast an anti-virus update to all or selected machines within your company. First, you would create a Generic Queue containing all the Execution Queues that represent machines to be used for this application. Next, you would choose “All Execution Queues” under the “Queue Selection” property on the Job Submission General tab. If you submitted this job it would then execute on every eligible queue/machine. This example assumes that the script file exists on all the execution machines and in the same location. What if that wasn’t true? Could you still use the “Job Broadcast” capability? The answer is YES! The solution lies in using the *Copy script…* facility.



Job File
File Name: c:\update.bat
Parameters:
☑ Copy script to execution machine

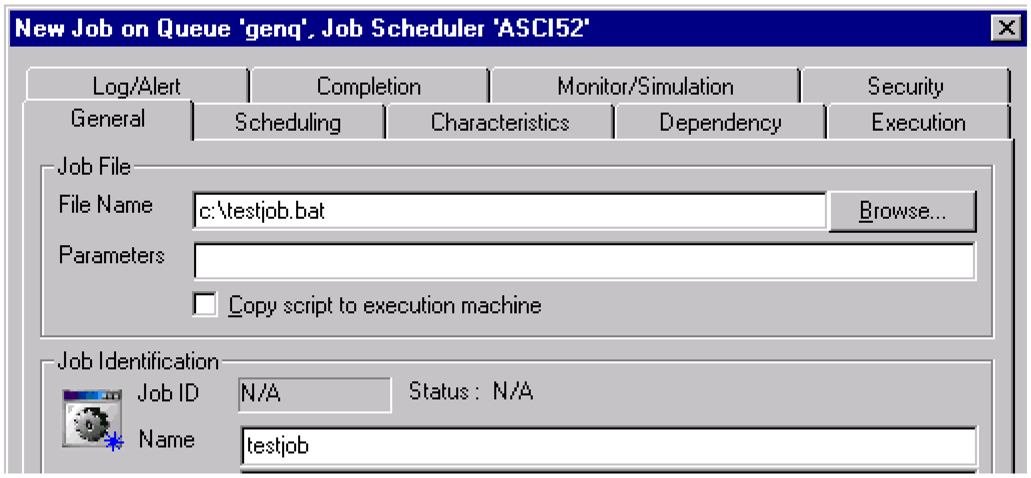
By enabling the *Copy script…* checkbox, the file *c:\update.bat* will be copied to each machine when the job is ready for execution.

7.3 Setting Up Your Jobs

While previous chapters have touched on all the properties associated with a Job, this section will provide more tutorial and advanced discussion on how to properly apply those properties. In particular with all the features present in the ActiveBatch product, this section will provide examples and ideas on how to use those features to your maximum benefit.

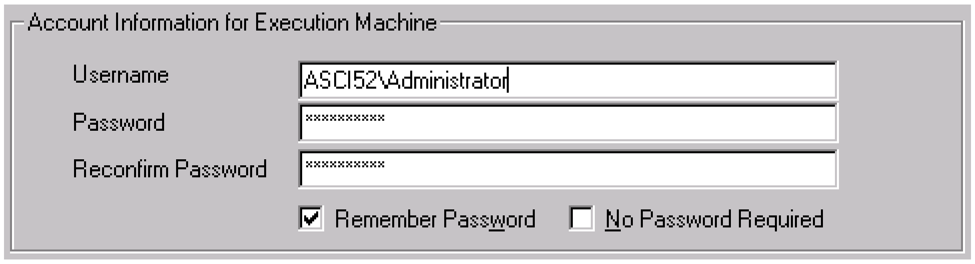
7.3.1 Job Property "Minimums"

ActiveBatch supports a myriad of Job properties. This section indicates the absolute minimum required for you to run a job. The figures that follow are all found on the various identified Job property tabs.



File Name: c:\testjob.bat
Parameters:
☐ Copy script to execution machine
Job ID: N/A
Status : N/A
Name: testjob

The above figure, found on the General tab, is the first property tab displayed. You must enter a filename specification. Believe it or not, sometimes new users enter this property incorrectly. The Job's Filename specification needs to be considered from the point-of-view of the ***execution machine*** and not the submitting client's machine. For example if the file *c:\testjob.bat* does not exist on the execution machine in the noted directory, the batch job will fail when executing. So when entering a local file you either need to know the file exists locally on the execution machine or you can enable the *Copy Script…* checkbox to ensure the file is copied from the submitting client's machine to the execution machine. You can also enter a UNC specification, however any share specifications must be accessible from the execution machine. Remember that use of the *Browse* button will examine files that are locally present on the submitting client's machine.


Account Information for Execution Machine

Username: ASCI52\Administrator

Password: **********

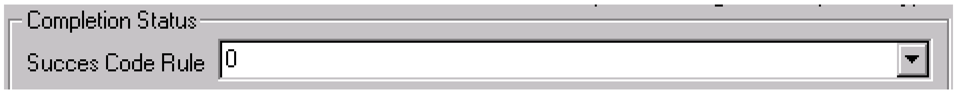
Reconfirm Password: **********

[✓] Remember Password [ ] No Password Required

The above figure, found on the Execution tab, shows the properties that associate a job with a security context that the job will execute under. The username must be valid on the execution machine and the password, entered twice to ensure accuracy, must match the specified username. ActiveBatch does not verify the username and password entered until the job is actually run on the target machine. For Windows 2000/NT systems, the username account must be granted the “Logon as a Batch Job” right. This right can be automatically granted by the ActiveBatch Execution Agent if enabled (see Section 3.2.1). To avoid re-entering the password each time, you can check the *Remember Password* checkbox. This causes ActiveBatch to conveniently store your password in an encrypted fashion using the same facilities as Windows 2000/NT itself.

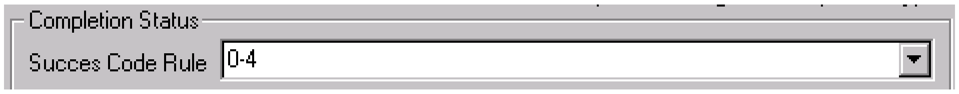
7.3.2 Success Code Rule

ActiveBatch has the ability to retrieve the job’s exit code after the job has executed and depending on whether the job has succeeded or failed, the product can take various actions. The problem is that no uniform method exists (especially over a heterogeneous set of platforms) for interpreting an exit status code. For Windows 2000/NT .BAT files, a typical success code is zero (0). Sometimes a user will set a code to one (1) to indicate success. For OpenVMS systems, an exit code that is odd is considered successful. To allow you to indicate what constitutes success for your batch job, ActiveBatch introduces the Success Code Rule.


Completion Status

Succes Code Rule: 0

This property, found on the Completion tab, indicates how a batch job step exit code is to be interpreted. The above figure shows the default success code rule. An exit code of zero (0) indicates that the job succeeded. You can enter multiple numbers or keywords by entering each value separated with a comma. A range of values can be indicated with a dash. For example,


Completion Status

Succes Code Rule: 0-4

shows a rule of zero through four meaning any of those exit codes indicate the job was successful. Keywords such as NTMSG (for those products which return NT message codes in which the severity is part of the message code) and ODD (for OpenVMS use) can also be specified.

7.3.3 Triggers

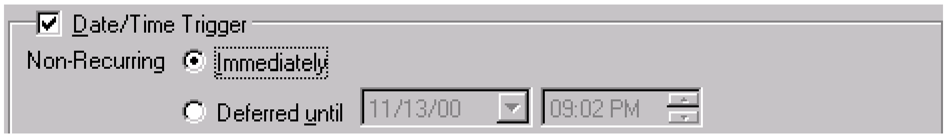
A **Trigger** indicates some type of event which when activated (or triggered) begins the process of running a job. ActiveBatch supports three (3) types of triggers: Date/Time based, Event based and ActiveBatch based. Proper use of these triggers can result in not just better performance or resource utilization but also enable the creation of applications that either weren't possible or easily created. This is particularly true of the Event based trigger that allows you to define your own events. All Triggers can be found on the Job Scheduling tab.

7.3.3.1 Date and Time Triggers

By far this class of trigger is the one most used for starting a job's execution. Four (4) types of sub-triggers are available: Immediate, Deferred, Interval and Schedule.

7.3.3.1.1 Immediate and Deferred

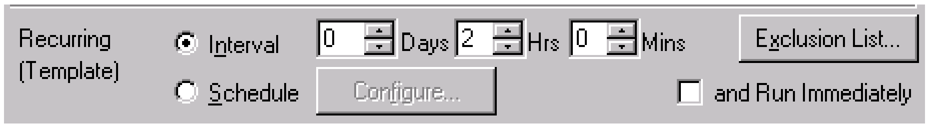
The simplest date and time trigger is **Immediate**. This means that the job is to start execution immediately. **Deferred** indicates that the job is to execute once at some future date and time. Immediate and Deferred triggers are referred to as "once only" triggers. This means that a job that uses either trigger will execute only once. ActiveBatch refers to "once only jobs" as "normal jobs". Much of the earlier Getting Started sections dealt with these simple triggers. Let's discuss some advanced date/time triggers.


☑ Date/Time Trigger

Non-Recurring: [✓ Immediately] Deferred until 11/13/00 09:02 PM

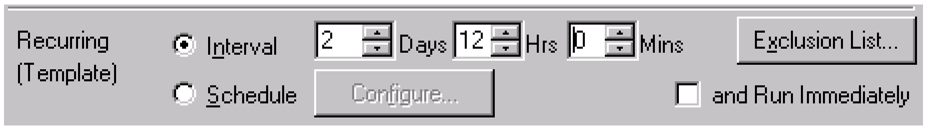
7.3.3.1.2 Interval

The **Interval** trigger is used when a job is to run based on a “delta” time period. For example, every 5 minutes or every 5 hours. The actual time of day may not be important but the time period is.



Recurring (Template) (●) Interval [0] Days [2] Hrs [0] Mins
( ) Schedule
[ ] and Run Immediately

This figure indicates an interval time of two (2) hours. Note that you can also add “days” to the interval such as:



Recurring (Template) (●) Interval [2] Days [12] Hrs [0] Mins
( ) Schedule
[ ] and Run Immediately

This example is 2 days and 12 hours.

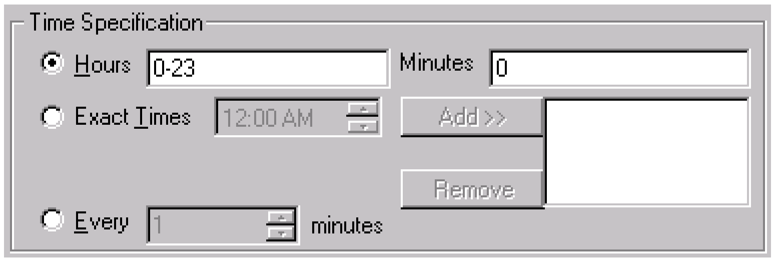
Jobs that can run more than once are known as **Template** Jobs. A template job produces an **Instance** when the template job is triggered for execution. This mechanism of templates and their instances allows greater flexibility in terms of controlling a single present job or future jobs.

7.3.3.1.3 Schedule

The **Schedule** trigger is used when a job needs to be more tightly controlled in terms of precisely running based on a date and time. A Schedule is uniquely named and a job can be associated with more than one schedule. A Schedule can also be shared and can support more than one job. The ability to associate multiple schedules with a Job means that you don’t have to spend a lot of time thinking about how to enter all schedule data into a single schedule.

A Schedule has several parts: Name, Description, Time, Date, Effective Date/Times and Security. This section will discuss the Time, Date and Effective Date/Times portions.

7.3.3.1.3.1 Time



Hours [✓] 0-23
Minutes 0
Exact Times 12:00 AM
Every 1 minutes

You have three (3) ways of entering time data into a Schedule:

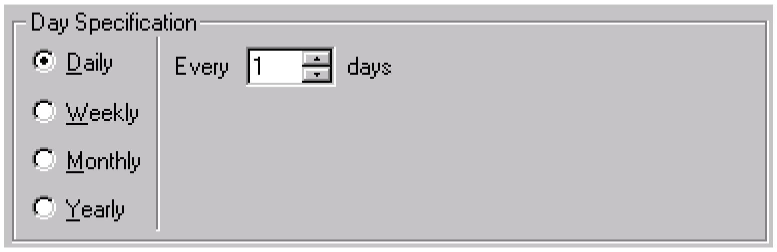
1. You can enter discrete hours and minutes that represent time of day. The hours and minutes that are entered are merged to produce time(s) of day. The default times above (hours: 0-23 and minutes: 0) produce an hourly schedule with the scheduled time trigger at the top of the hour (minutes: 0). For example, hours: 6,12 minutes: 0,15,30,45 would yield times of 6:00, 6:15, 6:30, 6:45, 12:00, 12:15, 12:30 and 12:45.
2. You can enter precise "exact times" by entering the time and then clicking the *Add* button. The time then appears in the window. To remove a time, select the time in the window and then click the *Remove* button.
3. You can enter a minutes value (1-59) which acts as an interval but is also aligned on the minute value entered. For example, 15 would mean every 15 minutes with triggers at :15, :30, :45 and :00.

The methods above are mutually exclusive (but remember you can always create another Schedule and associate it with the Job).

7.3.3.1.3.2 Date

You have four (4) ways of entering day/date data into a Schedule:

1. You can indicate that the schedule run "Daily" and be triggered every "n" days.



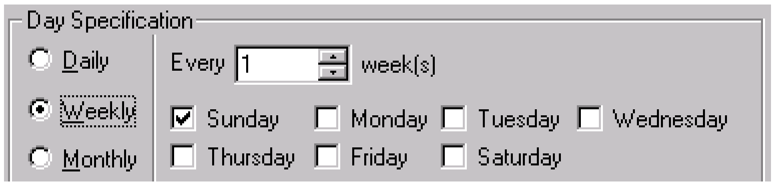
Day Specification

[✓ Daily] Weekly Monthly Yearly

Every 1 days

The above figure schedules a time period for every day.

2. You can indicate that the schedule be run weekly, on certain days, and be triggered every "n" weeks.



Day Specification

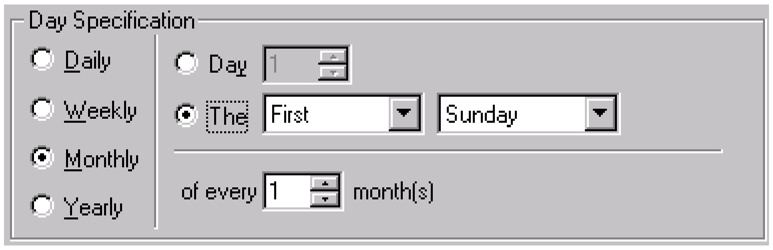
Daily [✓ Weekly] Monthly

Every 1 week(s)

[✓ Sunday] Monday Tuesday Wednesday Thursday Friday Saturday

The above figure schedules a time period every Sunday.

3. You can indicate that the schedule be run monthly, on either a day or relative day basis, and be triggered every "n" months.



Day Specification

Daily Weekly [✓ Monthly] Yearly

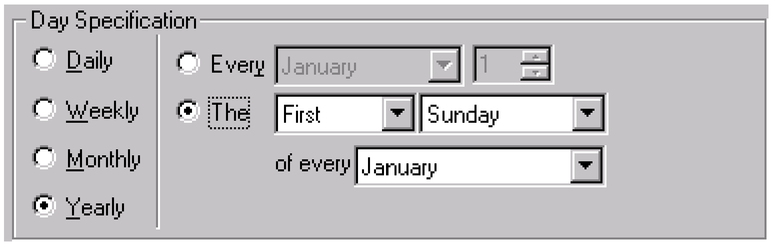
Day 1

[✓ The] First Sunday

of every 1 month(s)

The above figure schedules a time period for the first Sunday of each month.

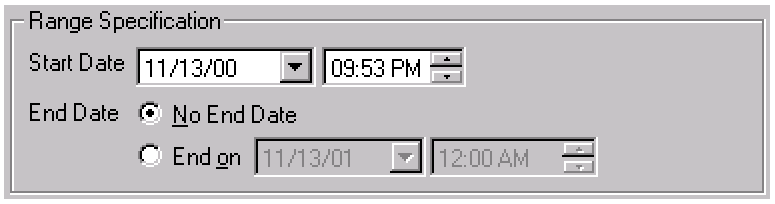
4. You can indicate that the schedule be run yearly on either a month/day or relative month/day basis.



[✓ Yearly]
[✓ The] First Sunday of every January

The above figure schedules a time period for the first Sunday in January.

7.3.3.1.3.3 Effective Date

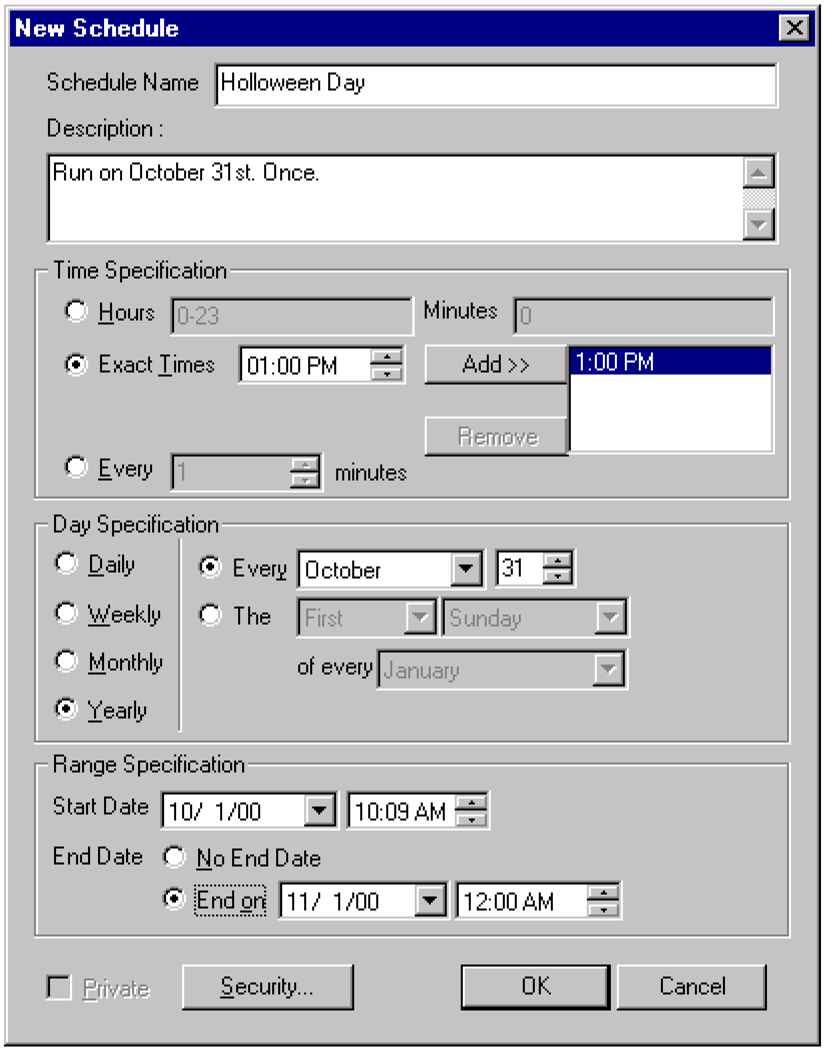


Start Date 11/13/00 09:53 PM
End Date [✓ No End Date]

This portion indicates the effective starting and ending date/time of the schedule. By default, no ending date/time is established. Since Schedules are by definition repetitive the effective starting and ending date/time provides a mechanism for performing a given day/time iteration a specific number of times.

7.3.3.1.4 Schedule Examples

Suppose you wanted to execute a job at 1:00pm on October 31, 2000… just once. How would you do it?



New Schedule

Schedule Name: Holloween Day

Description: Run on October 31st. Once.

Time Specification
- ○ Hours 0-23 Minutes 0
- ● Exact Times 01:00 PM — Add >> — 1:00 PM — Remove
- ○ Every 1 minutes

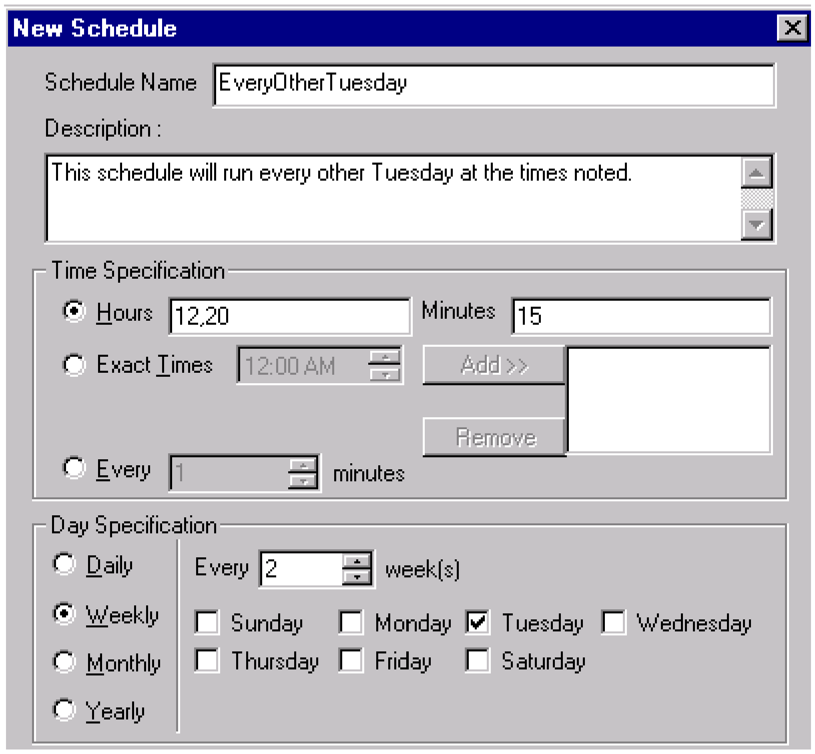
Day Specification
- ○ Daily
- ○ Weekly
- ○ Monthly
- ● Yearly
- ● Every October 31
- ○ The First Sunday of every January

Range Specification
- Start Date 10/ 1/00 10:09 AM
- End Date ○ No End Date
- ● End on 11/ 1/00 12:00 AM

☐ Private — Security... — OK — Cancel

The Schedule above is how you could do it. The secret is proper use of the Effective Ending time.

Let's try another. You'd like to schedule a job to run every other Tuesday at 12:15pm and 8:15pm.



New Schedule

Schedule Name: EveryOtherTuesday

Description: This schedule will run every other Tuesday at the times noted.

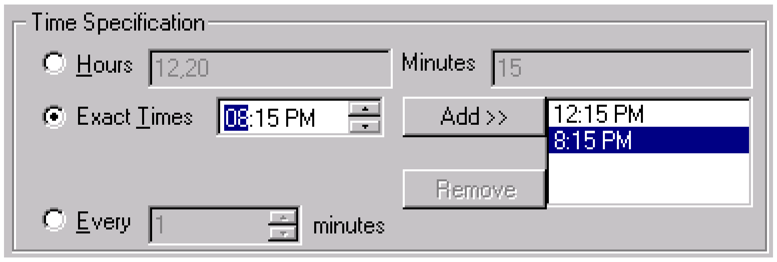
Time Specification
- (●) Hours: 12,20 Minutes: 15
- ( ) Exact Times: 12:00 AM
- ( ) Every: 1 minutes

Day Specification
- ( ) Daily
- (●) Weekly
- ( ) Monthly
- ( ) Yearly

Every 2 week(s)

- [ ] Sunday
- [ ] Monday
- [✓] Tuesday
- [ ] Wednesday
- [ ] Thursday
- [ ] Friday
- [ ] Saturday

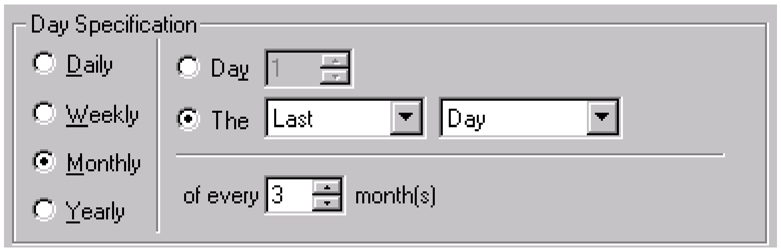
Note the use of Hours and Minutes in the Time Specification section. We also could have done it this way.



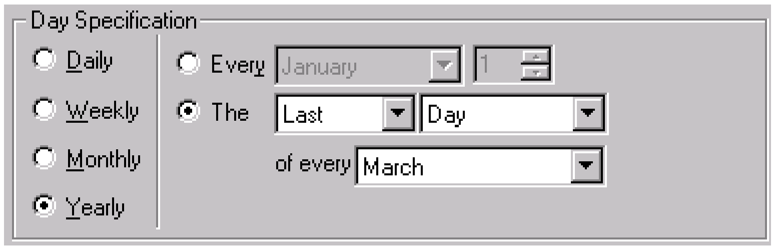
Time Specification
- ( ) Hours: 12,20 Minutes: 15
- (●) Exact Times: 08:15 PM
- ( ) Every: 1 minutes

Times list: 12:15 PM, 8:15 PM

Let's try one more. How would you create a Schedule that runs on the last day of every financial quarter? We'll we've got a few choices here.


Day Specification
Daily [✓ Monthly] Weekly Yearly
Day 1 [✓ The] Last Day
of every 3 month(s)

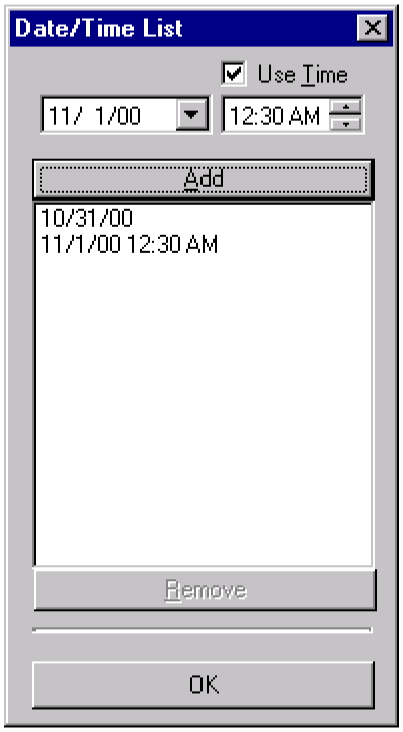
This would work nicely although since it is relative from the current month you need to start this schedule in the last quarter month (which you could easily do with an Effective Starting Day). You could also do each quarter month precisely like this.


Day Specification
Daily Weekly Monthly [✓ Yearly]
Every January 1 [✓ The] Last Day
of every March

You would then need to do the other quarter months, using additional schedules, as well.

7.3.3.1.5 Exclusion List

The exclusion list allows you to specify a date and/or specific times during a date when a date/time trigger should not occur. Hence the term "exclusion".



[✓] Use Time

11/ 1/00 12:30 AM

10/31/00
11/1/00 12:30 AM

The above figure shows that 10/31/00 (all day) and 11/1/00 at 12:30am only should be excluded from a Date/Time trigger.

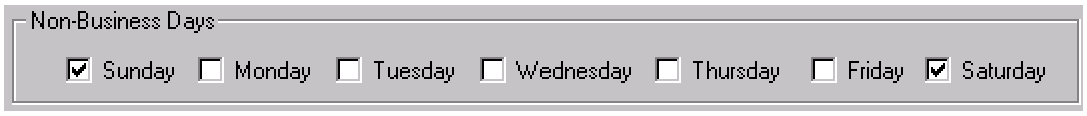
7.3.3.1.6 Time Zones

ActiveBatch allows you to specify which time zone to use when interpreting a time. The default time zone is the one used by the Job Scheduler. You can also select the Client's time zone or UTC (Universal Time Coordinated).

7.3.3.2 Business Date Processing (Calendars)

The Date and Time Trigger can be further refined through the introduction of a Business Calendar. By default, jobs which use date and time triggers run whenever those triggers occur regardless as to whether the date is a work day. For most administrative jobs, you will probably like that. For example, if a backup or defrag is to occur on a Saturday or Sunday that's probably OK. However, some business application jobs can only run on a working day. This concept of working or business day is supported by ActiveBatch through the creation of a Business Calendar. Business Calendars are uniquely named and you can have more than one Calendar in use.

The ActiveBatch Calendar supports two types of days: Non-Business and Holidays.

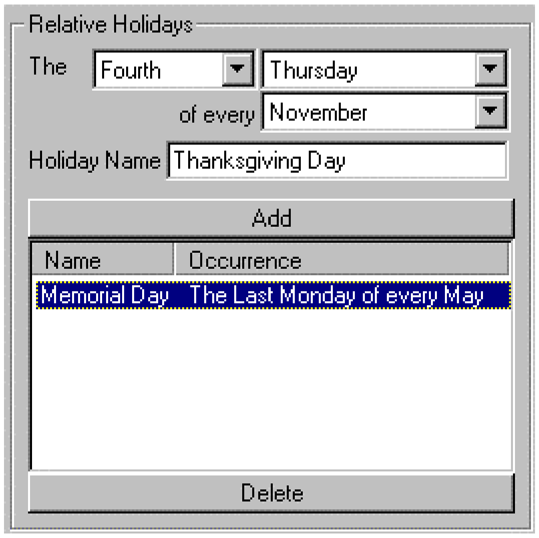

Non-Business Days

[✓ Sunday] Monday Tuesday Wednesday Thursday Friday [✓ Saturday]

As you can see above, non-business days are pretty straightforward. You simply check the days in which no business is conducted. ActiveBatch defines "non-Business" as days in which no business is typically conducted. This means the days which aren't checked are considered "Business Days" subject to Holiday exclusions.

ActiveBatch supports the concept of a "Holiday". A "Holiday" is defined as a day in which normal business would have occurred but doesn't because of the special day. Under this definition a "non-Business" day is never a "Holiday" and vice-versa. You can specify Holidays using any combination of two methods:

1. Relative Holidays. A holiday that is expressed in a relative manner, usually a day in a numbered week in a specific month. For example, Thanksgiving in the United States would be expressed as the Fourth Thursday in November.


Relative Holidays

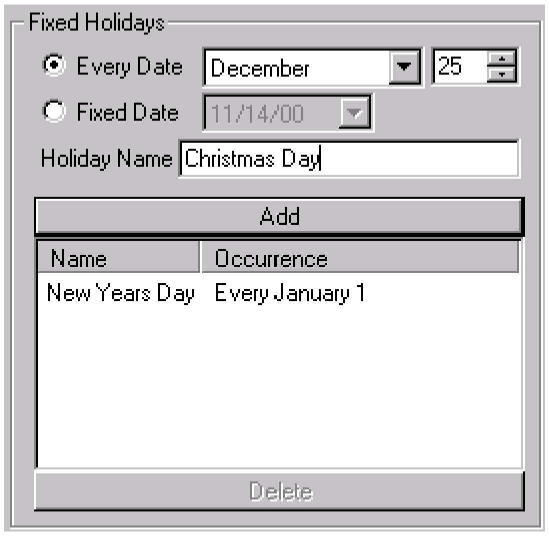
The Fourth Thursday

of every November

Holiday Name Thanksgiving Day

Name	Occurrence
Memorial Day	The Last Monday of every May

2. Fixed Holidays. This means a holiday that is based on a fixed date in a month. For example, New Years Day is always January 1. ActiveBatch also allows you to include a year as part of the specification in the event the holiday is temporary (or only applies to this year). A good example of this would be some sort of emergency (i.e. hurricane, etc) in which the office is closed in a non-scheduled manner.


Fixed Holidays

[✓ Every Date] December 25

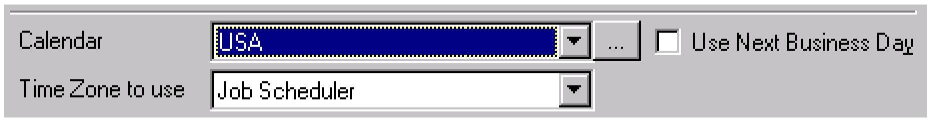
Fixed Date 11/14/00

Holiday Name Christmas Day

Name	Occurrence
New Years Day	Every January 1

To add a date as a Holiday make your date specification using one of the two methods above, enter a name for the Holiday (limited to twenty (20) characters) and click the *Add* button. This will add the date to the holiday window. You can use both methods to enter holidays. To delete a date, select the date you want to delete within the Holiday window(s) and click the *Delete* button. The date will disappear.

Once you create your Calendar(s) you need to then associate the calendar with the Job. This is done on the Job Schedule tab as shown below.


Calendar: USA
Time Zone to use: Job Scheduler

As a convenience you can click the "…" button to the right of the Calendar menu to view the selected calendar. When a Holiday falls on what would have been a business day you can indicate to ActiveBatch that you would like the job triggered for execution on the "next" business day. For example, let's say you have a job scheduled that runs every Monday and a Holiday happens to fall on a Monday. Since Holidays prevent the date/time specification from triggering, your job won't run on the Monday and would therefore run next week (7 days later). Clicking the *Use Next Business Day* checkbox indicates that you would like this job run on Tuesday which while not the normal day for running this job allows you to include an exception for this holiday.

7.3.3.3 Events

The Events based trigger in ActiveBatch allows you to create your own triggers based on something happening on a local or remote computer system. For example, let's say you wanted a job to run whenever a certain NT Event Log entry was written to, or whenever a customer FTP'd a file into a certain directory on your machine.

Two types of events are supported: WMI (WBEM) and System Startup.

Both event types support a common property. You may indicate that the event be disabled after it has been triggered using the *Trigger Once* property.

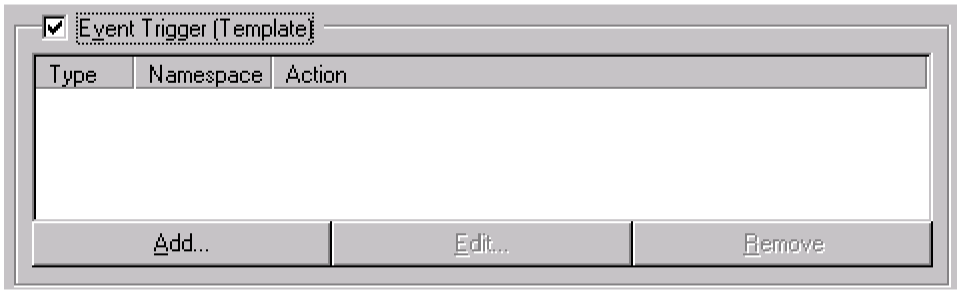
7.3.3.3.1 WMI Events

The engine behind events in ActiveBatch is WBEM. WBEM is the industry-wide term for Web Based Enterprise Management and was created by the industry Distributed Management Task Force (DMTF) that created standards for allowing information interchange between hardware and software vendors. The WBEM initiative lets vendors write Information/Event Providers that communicate with WBEM software using a database "schema" approach. This schema is referred to as the CIM (Common Information Model). Customers write WBEM consumers that ask WBEM providers for object information. The beauty of this approach is that a single extensible database is available for a wide and disparate variety of sources. WBEM provides a point of integration through which data from management sources can be accessed, and it complements and extends existing management protocols and instrumentation such as SNMP, DMI and

CMIP[4]. The Microsoft implementation of WBEM is known as Windows Management Instrumentation or WMI. WMI is freely available from ASCI for Windows NT V4 and 9x and is integrated into all Windows 2000 systems as part of the operating system installation.

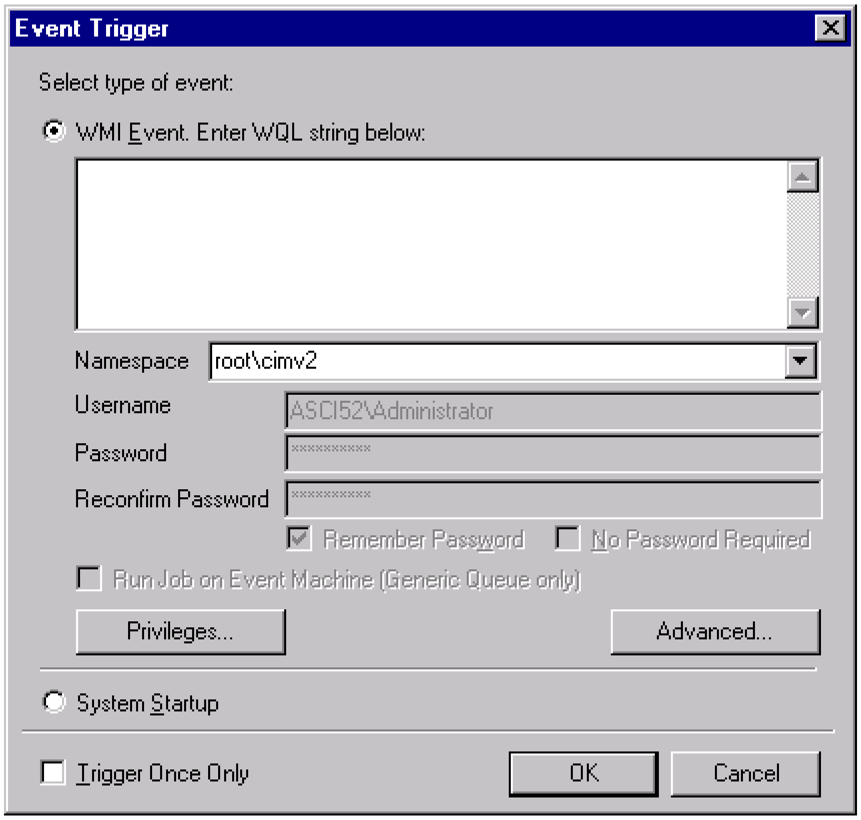
ActiveBatch fully supports WMI as it is both a Provider and Consumer of WMI information. Moreover, ActiveBatch allows you to create and request your own events. WBEM users do this using a syntax based on SQL named WQL. WQL is a cross-platform WBEM language for requesting WBEM services. With WQL almost every hardware and software object can be queried for event usage. For example, suppose you wanted a batch job to begin execution when the free space on drive C: dropped below 30 percent. In the "old" days you would have written a batch job based on a time trigger which periodically ran and checked for the above condition. Now you can directly create an event in which ActiveBatch will only start the job when the condition occurs.

Before we examine Event based triggers in greater detail let me remind you that you will need WMI installed on at least the Job Scheduler machine for any of these capabilities to be used.



The above figure can be found on the Job Scheduling tab and needs to be enabled so you can add an event. Clicking the *Add* button causes the dialog on the following page to appear.

[4] This sentence was extracted from the Microsoft Windows Management Instrumentation Provider Programming White Paper, 1999.


Event Trigger

Select type of event:

[✓ WMI Event. Enter WQL string below:]

Namespace: root\cimv2

Username: ASCI52\Administrator

Password: **********

Reconfirm Password: **********

[✓ Remember Password] [ ] No Password Required

[ ] Run Job on Event Machine (Generic Queue only)

( ) System Startup

[ ] Trigger Once Only

This dialog provides the ability to request the creation of a WMI event. As stated previously, the language for requesting WMI queries is WQL. While this manual cannot be an exhaustive reference to the WQL language some aspects can be discussed.

All WQL event query statements begin with the word SELECT. A typical WQL event query looks like this:

*SELECT * FROM eventclass {WITHIN | GROUP | HAVING}*

WHERE property Operator value

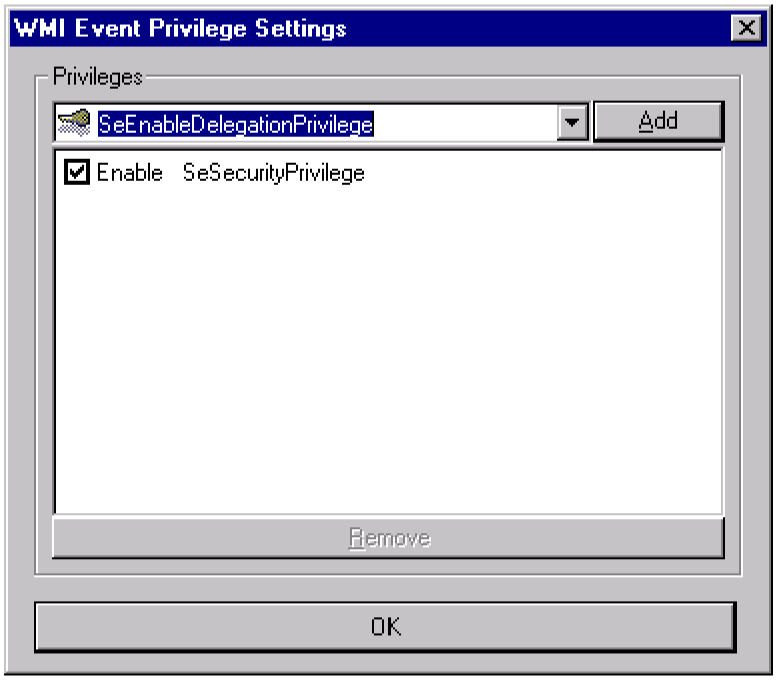
For example an event query concerning freespace on drive C: might look like this:

```
SELECT * FROM __InstanceModificationEvent WITHIN 60
        WHERE TargetInstance ISA “Win32_LogicalDisk”
        AND     TargetInstance.DeviceId = “C:”
        AND     TargetInstance.Freespace < 1000000
```

This event query checks every 60 seconds as to whether logical disk device "C:" freespace drops below 1 million bytes. If so, an event would be generated which would trigger execution of the batch job. While certain keywords within WQL are optional your use of WMI within ActiveBatch will be improved if you follow these guidelines:

1. Unless you are absolutely certain you should use the WITHIN keyword to base all events on a period of time. WMI will enforce the grouping of certain queries particularly if an Event Provider for the property query is not available. Otherwise an excessive amount of CPU would be used.
2. When you expect a lot of the same events to be generated as a result of an event query, you should consider using the GROUP keyword to aggregate these events into a single event notification.

WMI operates on a namespace basis. A namespace can be local or remote. To enter a remote namespace simply add a machine name using UNC style syntax. For example: \\asci52\root\cimv2 would connect to machine "asci52\root\cimv2" namespace. Local connections are performed under your Windows 2000/NT security context. You cannot change that context which is why the account information is grayed out. If you specify a remote machine you will be required to specify security information with which to create the event query. Some event queries require special privileges or rights. If you need to assert or deny a privilege/right to a provider for your event query, simply click the *Privileges* button.

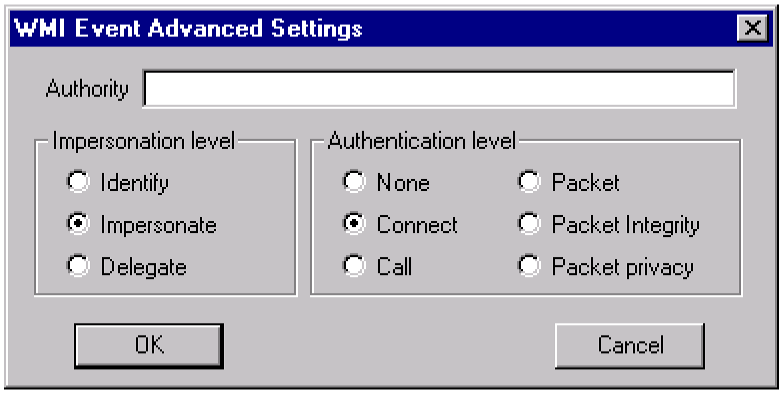

WMI Event Privilege Settings

Privileges

SeEnableDelegationPrivilege

[✓] Enable SeSecurityPrivilege

The above dialog shows the *SeSecurityPrivilege* enabled. This might be used for an NT Event Log query that requires that right to be asserted to the WMI Event Log Provider on your behalf.


WMI Event Advanced Settings

Authority

Impersonation level: Identify [✓ Impersonate] Delegate

Authentication level: None [✓ Connect] Call Packet Packet Integrity Packet privacy

The above dialog is for advanced users only and allows for different kinds of security connections. The default settings shown above are typically what you will want.

Jobs which are submitted to Generic Queues may also indicate that the job be run on the first Execution Queue which matches the machine that generated the event. If no Execution Queues match the event machine the job will not run.

7.3.3.3.2 System Startup Events

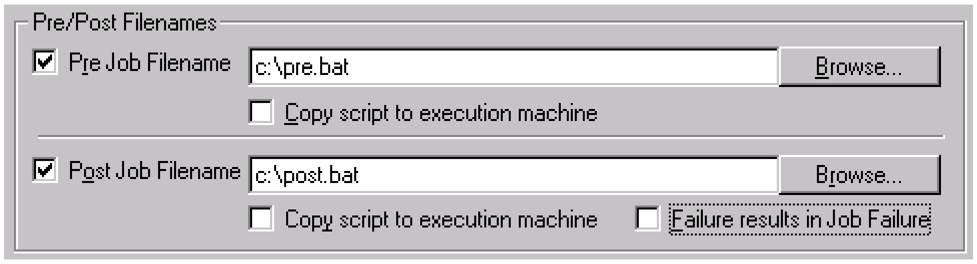
An event marked for System Startup means that the job will reside on the submitted Execution Queue and be triggered for execution only when the Job Scheduler machine is started. System Startup events do not require WMI to be installed.

7.3.3.4 ActiveBatch Trigger

This type of trigger provides support for several ActiveBatch features. The Trigger command, Batch Start Release and Alert Job Release all require this trigger to be enabled.

7.3.4 Pre and Post Job Steps

Most Batch Job Management systems only support the execution of a single job step. This means that all requirements and dependencies of the job must be part of the script or program that is to be run. With ActiveBatch you can define pre and/or post job steps which will execute before and then after the main job step.

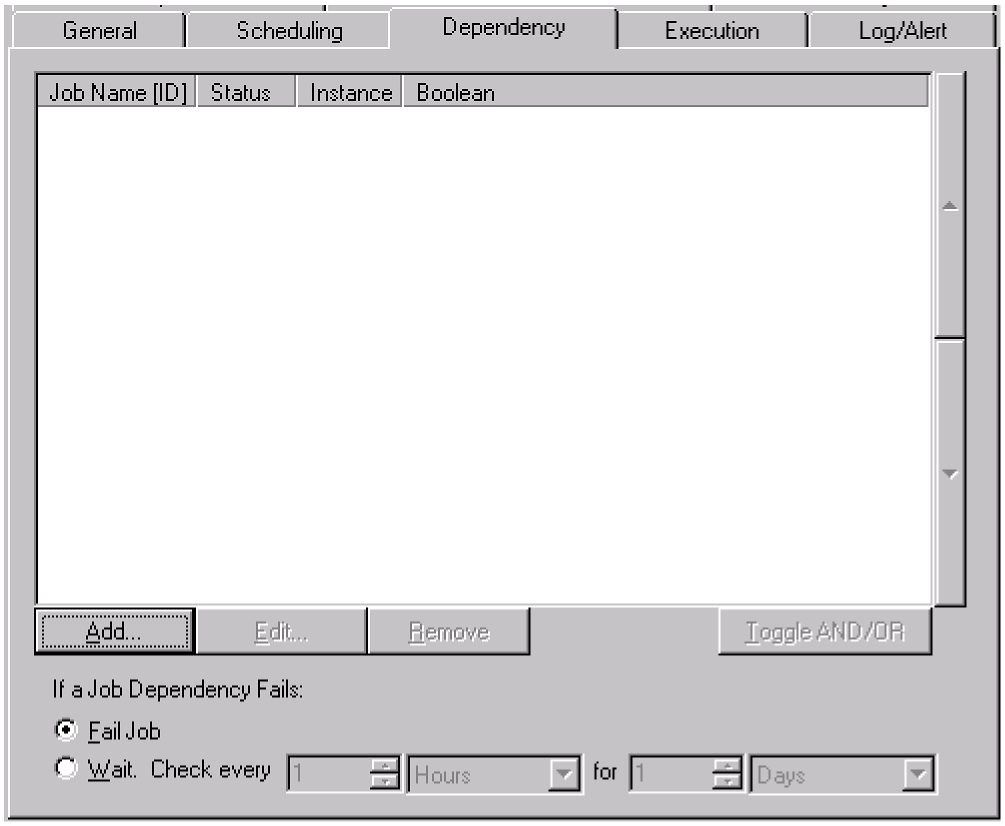

Pre/Post Filenames
[✓] Pre Job Filename: c:\pre.bat
[ ] Copy script to execution machine
[✓] Post Job Filename: c:\post.bat
[ ] Copy script to execution machine
[ ] Failure results in Job Failure

The "Pre" job step is intended to be used to establish the proper running environment for the main job step. This usually means moving whatever files might be necessary for the main job step to properly operate. For example, let's say you need to run a job which updates various machines with a new database. Rather than creating an overly complex main job step you could write a Pre step that copied files from the client's machine to the execution machine (using the ABAT_CLIENTMACHINE environment variable), run the main job step and then, using a Post step, check the severity of the main job step's exit status and move the new files back to the client's machine. All job steps are subject to restart recovery and checkpointing can be used. Many environment variables (or logical names) are created to provide additional ActiveBatch information to all the job steps. As ActiveBatch is COM based, depending on the scripting language you use, your job steps could also inquire of ActiveBatch directly concerning any Job properties. You can create jobs using any or all of the pre, main and post steps (of course all jobs must have a main step). When you write your Post step script, that script should check the ABAT_SEVERITY variable to determine whether the job succeeded or failed. Please note that the Job's Success Code Rule governs all job steps. Just as the main step can be copied to an execution machine, the other steps can be individually copied by ActiveBatch as well.

Note: If the Pre job step fails, the main step will not be executed and the job is considered to have failed. If the Main job step fails, the Post job step will still be executed (this is handy for performing various cleanup operations). Regardless of how the Post step executes the overall Job state is governed by the Main step *unless* you click the *Failure results in Job Failure* checkbox for the Post job step. As with all job steps any file specifications entered are considered local or otherwise accessible from the execution machine's point-of-view (unless the *Copy Script…* checkbox is enabled).

7.3.5 Job Dependencies

One of the nice features of ActiveBatch is the ability to create a group of jobs that are related in some fashion. Typically many jobs need to run sequentially depending on how the previous job performed. In ActiveBatch we refer to these as *Job Streams*. A Job Stream can be as small as 2 or as large as several hundred jobs (there is no architectural limit). When Job B can only be executed depending on how Job A performs we refer to that as a *Job Dependency*. A Job can be dependent on one or more other jobs. In addition the nature of the dependency can be conditional. By that we mean that the job's completion status can be required as being successful or merely completed. When a Job has a dependency that is not met, we refer to that as a *Dependency Failure*. When a Dependency Failure occurs you can either fail the job stream or wait some period of time for the dependency to be repaired.



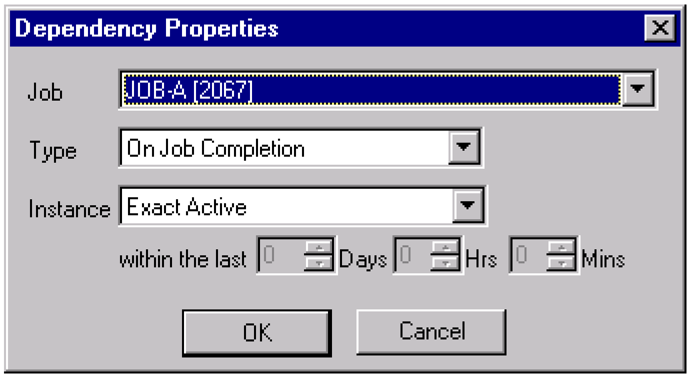
Job Name [ID]	Status	Instance	Boolean

If a Job Dependency Fails:

- [✓ Fail Job]
- Wait. Check every 1 Hours for 1 Days

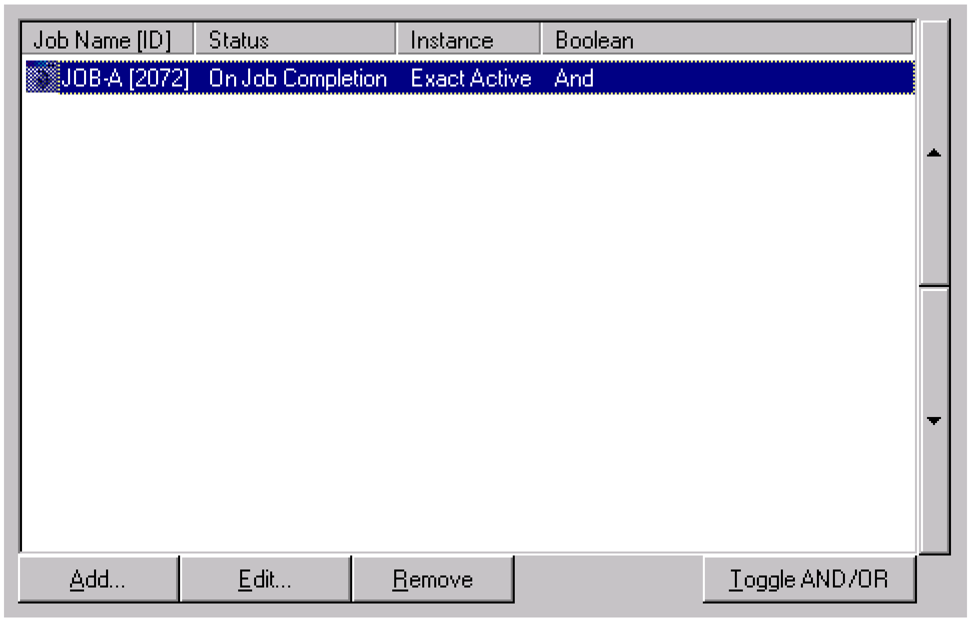
The dialog above can be found on the Job Dependency tab. Click the *Add* button to add a job dependency.

Let's assume a simple two (2) job stream: JOB-A and JOB-B, where JOB-B is dependent on JOB-A. The dialog below is displayed when the *Add* button is clicked on the Job Dependency tab.



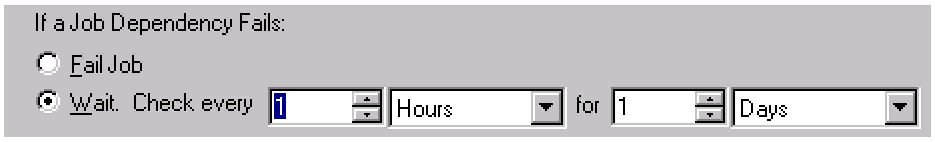
Job: JOB-A [2067]
Type: On Job Completion
Instance: Exact Active
within the last 0 Days 0 Hrs 0 Mins

The *Job* field is a pulldown consisting of all jobs that are eligible for job dependency checking. The *Type* field indicates the condition of the dependency. Valid types are *On Job Completion* or *On Job Success*. On Job Completion means that the job must merely complete. On Job Success is the more stringent condition and indicates the job must have succeeded for the dependency to be met. The *Instance* field indicates the time period scope of the previous job run. *Exact Active* is the most stringent value and indicates that the previous job run is either the one currently active or the last "regularly" scheduled job run. *Last Completed* indicates Exact Active but also allows the time period to expand to include the last completed job run. You determine the time period for the last completed job by indicating the time in days, hrs and mins. If a job ran and completed in that time period according to *Type* that job run is considered. The last instance value is *All Instances* which means if the job ever ran it will be considered (again according to *Type*).



Job Name [ID]	Status	Instance	Boolean
JOB-A [2072]	On Job Completion	Exact Active	And

The confirmed dependency appears in the dialog above. You'll notice that in addition to the original information a new column *Boolean* also appears. ActiveBatch supports boolean operations on job dependencies. This means that you can say "Dependency 1" AND "Dependency 2" OR "Dependency 3". You can toggle the boolean operator for a given dependency by selecting the job dependency and then clicking the *Toggle AND/OR* button. You can also change the order of the job dependencies by selecting the dependency and then clicking on the UP and DOWN arrow buttons on the far right side of the dialog tab. By the way, if one dependency is present, as it appears above, don't worry about the stray AND boolean.



If a Job Dependency Fails:

Fail Job [✓ Wait.] Check every 1 Hours for 1 Days

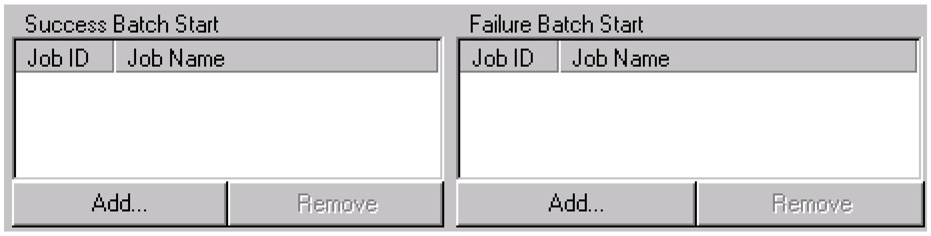
The radio buttons above determine what you want to do should one or more job dependencies fail. Your two action choices are:

1. **Fail Job**. This is pretty straightforward. ActiveBatch will fail out the job due to the dependency failure.
2. **Wait**. This action indicates that ActiveBatch is to wait for a period of time (you can also indicate how often to check). During this period if the dependencies are resolved the job is allowed to continue.

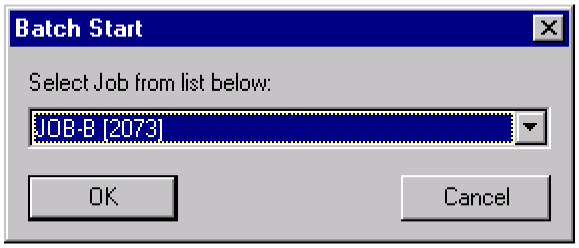
7.3.6 Batch Starts

Job Dependencies are essentially predecessor conditional operations in which a previous job must complete to some level. Batch Starts take the form of children triggers which are activated depending on how the job did (failure or success).

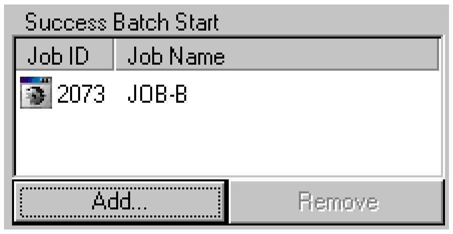
Using our earlier JOB-A and JOB-B stream. We've already said that JOB-B is dependent on JOB-A but wouldn't it be cool to have JOB-A trigger JOB-B to start execution if JOB-A succeeded. That's essentially one of the thrusts of the Batch Start facility.



The area above comes from the Job Completion tab. When JOB-A completes you can indicate which job(s) are to be triggered if JOB-A succeeds or fails.



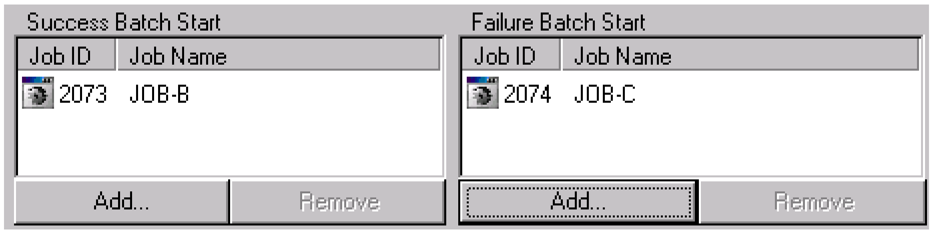
Clicking the *Add* button under the Success Batch Start area, causes the above job pulldown list to appear. As you can see JOB-B is already listed. By clicking OK the following appears:



Job ID	Job Name
2073	JOB-B

This means that should JOB-A succeed, JOB-B will be triggered for execution. JOB-B must, of course, have ActiveBatch Triggers enabled (see Section 7.3.3.4).

An interesting aspect of Batch Start is the ability to trigger batch jobs based on a job's failure. If JOB-A failed perhaps we would want to trigger JOB-C which might take corrective action and re-run JOB-A.



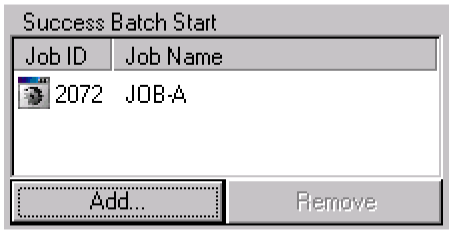
Success Batch Start

Job ID	Job Name
2073	JOB-B

Failure Batch Start

Job ID	Job Name
2074	JOB-C

Like this. This is JOB-A's Batch Start grid.



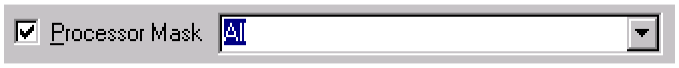
Success Batch Start

Job ID	Job Name
2072	JOB-A

This is JOB-C's Batch Start window.

7.3.7 Multi-Processor Affinity

Multi-Processor systems are rapidly becoming the norm. ActiveBatch provides a major facility for associating processors with a given batch job. This allows you to deploy critical CPU resources where you best need them.



☑ Processor Mask: All

This field, found on the Job Execution tab, allows you to indicate which processors are to be used to execute this job. The default is ALL. You fill out this field by indicating a range of values from zero (0) through the number of CPUs you have (minus one) as to which processors should be used. For example, 0, 5-7 means that CPU 0, 5, 6 and 7 are to be used. Please note that some operating systems won't allow you to deploy CPU 0 as part of a processor set.

7.3.8 Job Operational Features

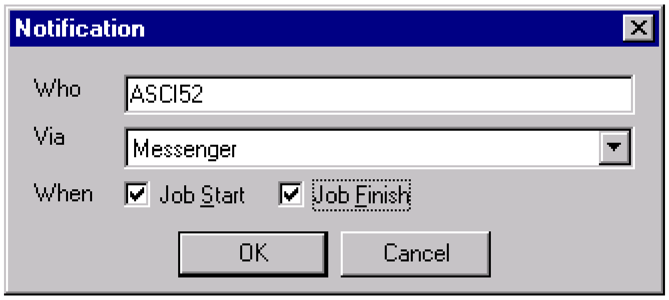
Several features are available for all ActiveBatch jobs to use.

1. **Job/Queue Priority**. This field found on the Job Execution tab allows a user to specify a priority when queuing a batch job. The default value is ten. The higher the value, the higher the priority. You should consider specifying both lower and higher priorities when you have jobs which can either wait or are more time critical than other normal jobs.
2. **O/S Priority**. This field, which should not be confused with Job Priority, controls the execution priority of the job within an operating system context. Windows 2000/NT uses keywords which indicate the priority class of execution. Many other operating systems use priority numbers (usually 0 to 31), with the higher number indicating a higher priority.
3. **On Job Active**. This field allows you to determine whether this template job can tolerate any other simultaneous copies of itself already running. Three (3) action choices are possible: *Skip* (default) indicates that if the job is already running, this new attempted instance should not be started. *Run Multiple* indicates that multiple copies of the job are allowed to execute. *Wait* indicates that ActiveBatch should wait for a period of time (user specified) to allow the already running instance to complete. If so and within the time period, the new instance is allowed to start. If the time period expires the new attempted instance is skipped.
4. **Job Holds**. ActiveBatch supports the automatic Hold of a Job as part of the Job's properties. Three (3) conditions are supported in which a Job is automatically held.
   (a) Hold on Submit. This property, found on the Job Scheduling tab, allows a user to hold a job on submission. This feature allows you to enter a job into the system without fear of execution until you are ready (in which case you would Release the job).
   (b) Hold Job on Error/Abort. This property, found on the Job Completion tab, allows a user to hold a job when the job completes in error or is aborted. The Hold is intended as a visual signal that a problem may have occurred. Note that when the job is released, the job state will change to a "Completed" state.
   (c) Hold Job on Completion. This property, found on the Job Completion tab, allows a user to hold a job when the job completes. As with "Hold Job on Error…" this Hold is intended as a visual signal that a problem may have occurred. Note that when the job is released, the job state will change to a "Completed" state.
5. **Terminate All Child Processes**. This feature indicates that any child processes created by the batch job are to be terminated when the main batch job exits. Different operating systems offer varying support for this feature. Windows NT V4 does not offer a foolproof method of terminating all processes although ActiveBatch will attempt to find and terminate processes started by the job. Windows 2000 and many other operating systems offer more integrated support for this feature.

7.3.9 Notifications and Alerts

ActiveBatch provides two (2) facilities to assist you in detecting when jobs start/finish and/or encounter other ActiveBatch states: Notifications and Alerts.

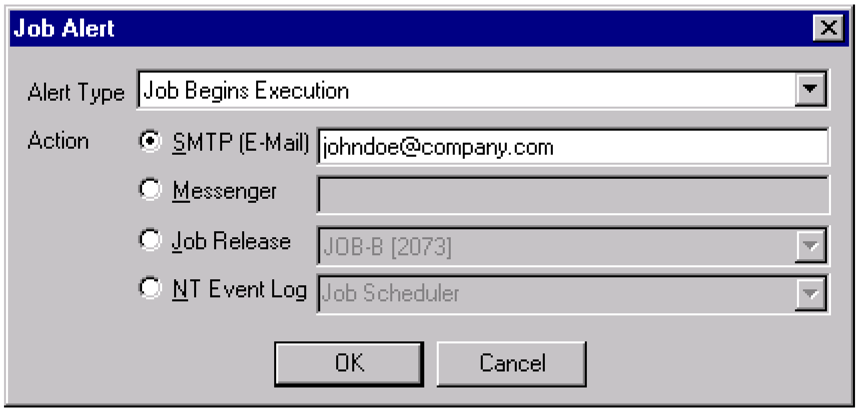
Notifications are used when the interest in a job starting or ending is purely informational and not critical. Notifications are limited to a message indicating whether the job is starting or finishing. NT Messenger or SMTP e-mail is used to send the notification.



Who: ASCI52
Via: Messenger
When: ✓ Job Start ✓ Job Finish

Alerts are used to pinpoint ActiveBatch states and take some level of action. Alerts can choose any combination of actions from:

1. Windows NT Messenger Service.
2. SMTP E-mail.
3. NT Event Log.
4. Batch Job Release Trigger.

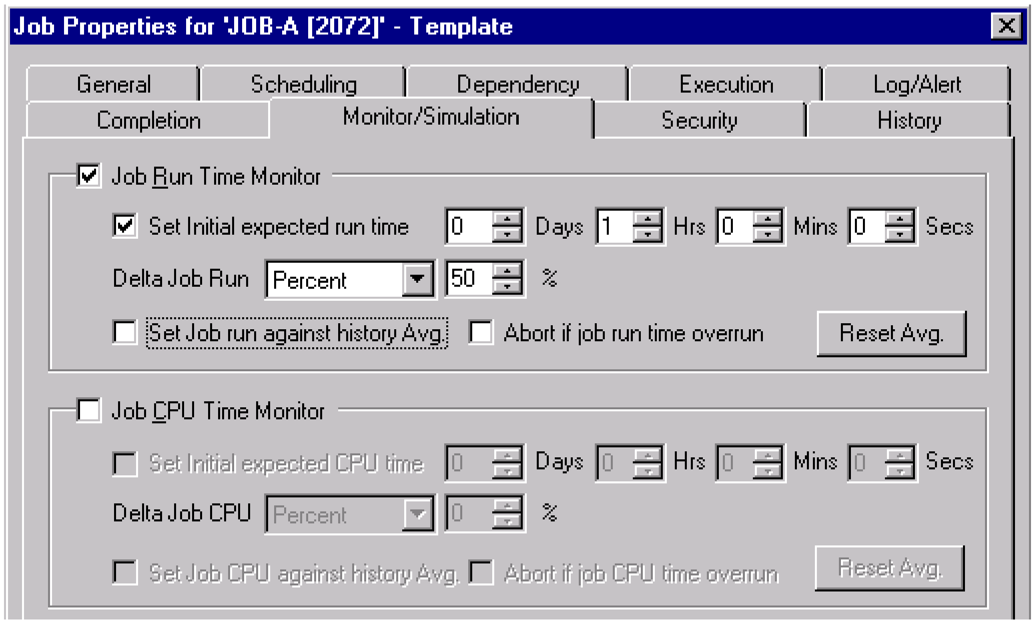


Alert Type: Job Begins Execution
Action: ● SMTP (E-Mail) johndoe@company.com
○ Messenger
○ Job Release JOB-B [2073]
○ NT Event Log Job Scheduler

The first three actions provide alert capability in the true sense of the word. The last action allows you to take whatever additional action you require.

7.3.10 Job Monitoring

One of the advanced uses of ActiveBatch is for the product to monitor your job in terms of elapsed and CPU time usage. If a job enters a loop it may remain in that state for hours before someone realizes the situation. The problem, of course, is that a lot of time has been wasted.

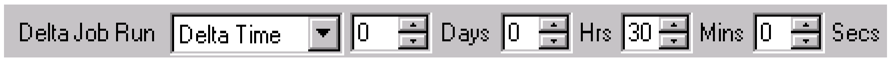

Job Properties for 'JOB-A [2072]' - Template

General | Scheduling | Dependency | Execution | Log/Alert
Completion | Monitor/Simulation | Security | History

[✓] Job Run Time Monitor
[✓] Set Initial expected run time: 0 Days 1 Hrs 0 Mins 0 Secs
Delta Job Run: Percent 50 %
[ ] Set Job run against history Avg. [ ] Abort if job run time overrun — Reset Avg.

[ ] Job CPU Time Monitor
[ ] Set Initial expected CPU time: 0 Days 0 Hrs 0 Mins 0 Secs
Delta Job CPU: Percent 0 %
[ ] Set Job CPU against history Avg. [ ] Abort if job CPU time overrun — Reset Avg.

The display above can be found in the Job Monitor/Simulation tab. Two sections describe the job monitoring capabilities of ActiveBatch. The first deals with elapsed or run-time monitoring. The second deals with CPU time monitoring. Since both sections are essentially the same in terms of concepts we will focus on elapsed/run-time monitoring. To enable Run-Time monitoring you can simply click on the checkbox. Typically you will indicate the initial expected run-time using the appropriate fields. You can also indicate whether you will allow a plus/minus variance. This is known as “Delta Job Run”. The delta can be expressed as either a percent or an actual time specification. In the example above, the initial expected time is set to one (1) hour. The delta is set to fifty (50) percent. This means that 1 hour +/- 30 minutes is the acceptable time range for the elapsed time. The same delta could also have been set as:


Delta Job Run: Delta Time 0 Days 0 Hrs 30 Mins 0 Secs

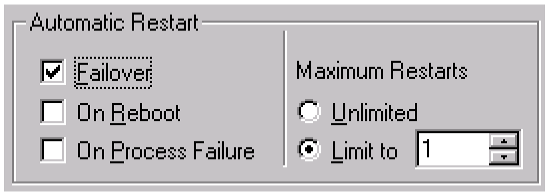
The “plus/minus” aspect of job monitoring is quite important since an underrun in elapsed time may be as serious as an overrun. ActiveBatch allows you to establish static or dynamic job monitoring. To establish dynamic job monitoring, you simply click the *Set Job run against history Avg.* checkbox. This causes the Job Scheduler to average the elapsed times of the previous runs.

The advantage to this approach is that less management (manual intervention) of the job monitoring feature is required. After all many batch jobs for successful businesses take longer to run over time. To reset the average simply click the *Reset Avg.* button.

With Job Monitoring enabled, underrun or overrun elapsed times will be treated as possible alerts. However, you can also take action if an overrun occurs outside the alert capability. The *Abort if job run time overrun* checkbox controls whether ActiveBatch should take an active stance when an overrun is detected. With this checkbox enabled, ActiveBatch will abort a batch job that exceeds both the expected and delta time periods. The specific error code ABAT_EXCRUNTIME (or ABAT_EXCCPUTIME) is used to abort the job.

7.3.11 Restart and Job Checkpointing

ActiveBatch supports Job Restart due to machine or software failure. The Restart capabilities for a Job can be found in the Job Completion tab.


Automatic Restart
- [✓] Failover
- [ ] On Reboot
- [ ] On Process Failure

Maximum Restarts
- ○ Unlimited
- [● Limit to] 1

Three (3) types of Restarts can be requested: Failover, Reboot and Process Failure. When more than one action is specified the following order is used: Failover, Reboot and On Process Failure. Regardless of Restart action you can limit the total maximum number of restarts to a specific value.

When a Job is restarted, each step is re-executed. Jobs can determine whether they have been restarted by examining the ABAT_RESTART environment variable. A YES value indicates the job has been restarted. A NO value indicates the job has been normally started.

7.3.11.1 Failover

Job Failover restart is available to jobs that have been submitted to a Generic Queue. When an execution machine fails, the job is moved from the failed Execution Queue back to the Generic Queue where it can be re-dispatched. Failover is not available when a Job has been submitted *directly* to an Execution Queue. While Failover is typically used on Cluster based systems it need not necessarily be so. This is especially true if the data or other factors important to the job are still available on other remaining machines.

7.3.11.2 On Reboot

On Reboot restart is available to all jobs regardless of how the job was submitted. When a job is marked for “On Reboot” restart, the failed job enters a “Pending (Waiting for Reboot)” state. When the machine is once again available a check is made to determine whether the job is still known and active by the Execution Agent. If so, the job re-enters the “Executing” state. If the job is not known, the Job is restarted. It is important to note that the job will wait an unlimited time for the execution machine to become available.

7.3.11.3 On Process Failure

On Process Failure indicates that should the job fail (for any reason other than an abort) a restart of the job should be performed. If you enable this feature you should consider a low maximum restart limit to avoid a loop where the job continually fails.

7.3.11.4 Checkpoint

The preceding sections discussed job restart and re-running a failed job. While that feature is important and useful in and of itself the ability to pickup where the job left off is highly desirable. ActiveBatch offers a language independent checkpoint utility which allows a job to checkpoint whatever state information may be needed to restart and pickup at or near the last processing point. The ActiveBatch utility program **AbatChkPt** is a console based application that takes a single argument; Checkpoint Data. For example:

```
      abatchkpt “Processing Step 4”
```

The string “Processing Step 4” would be stored along with the job.

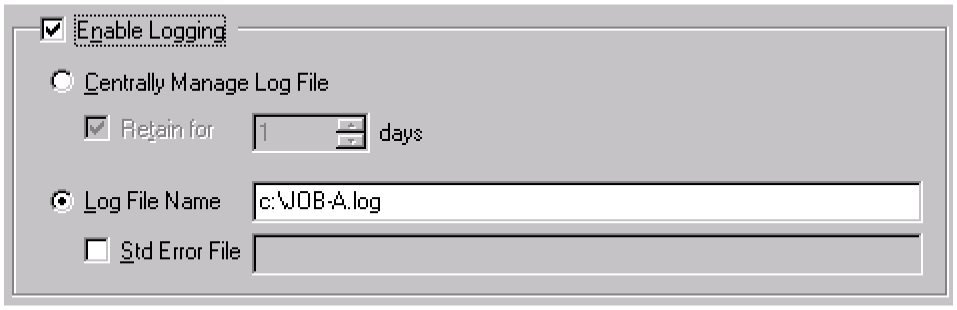
That data is sent to the Job Scheduler and saved along with the job instance. In the event a restart is performed several environment variables are provided to determine and then retrieve checkpoint information. ABAT_CHECKPOINT is a boolean variable which indicates whether valid checkpoint data is available. ABAT_CHECKPOINTVAL is a variable which contains the last checkpoint information. ABAT_CHECKPOINTTIME is a variable that contains the last date and time a checkpoint was taken. Please note that you can only invoke the AbatChkPt program within an active batch job script.

7.4 Troubleshooting

ActiveBatch was designed to assist Administrators and users and help track down batch job problems. The sections that follow discuss some of the features that can be used to determine why ActiveBatch did or didn't execute a job as well as determining why a job didn't perform properly.

7.4.1 Job Output Logging

ActiveBatch jobs have great flexibility in the area of logging. Logging in the ActiveBatch context means the capture of a job's standard output and error streams. These streams (usually found under different names depending on the operating system) are available for every job.


☑ Enable Logging
○ Centrally Manage Log File
☑ Retain for 1 days
● Log File Name c:\JOB-A.log
☐ Std Error File

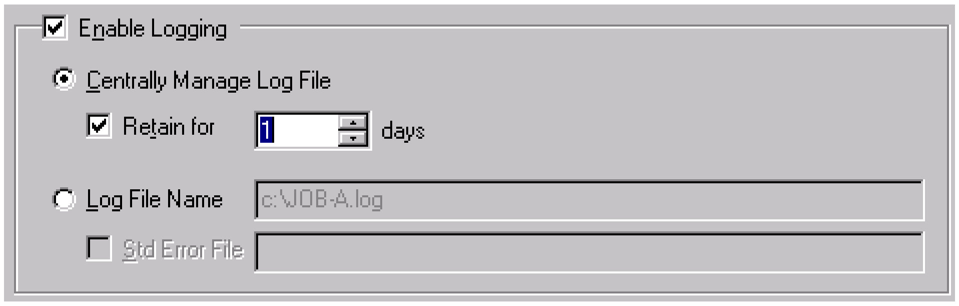
7.4.1.1 User Managed Logging

The example above is the default logging properties for a job named JOB-A. ActiveBatch, by default, uses the same location and filename as the main job step. The extension .LOG is added. When JOB-A is started all data to the standard output and error streams will be directed to "c:\JOB-A.log". If you don't like the defaults you can always change the entire file specification. If the job is a template job and is subject to multiple runs then the file specification is used above *except* that the job-id, date and time are also appended to the filename portion. The intent of this is to make the file specification unique such that other instances of the same job don't overwrite each other. Clicking the checkbox *Std Error File* causes the standard error stream to be directed to a separate file that is different than that used for standard output.

Logging in this fashion is referred to as *User Managed*. That is because you are required to manage the files (purge them, etc).

7.4.1.2 Centrally Managed Logging

Another choice is for ActiveBatch to manage the log files. After all many users don't care about the batch job log unless the job fails. By clicking the radio button *Centrally Manage Log File* you enable this facility. This means that the location and name of the log file is completely left to ActiveBatch. So what do you gain?



☑ Enable Logging

● Centrally Manage Log File

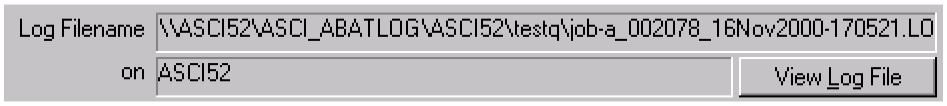
☑ Retain for 1 days

○ Log File Name c:\JOB-A.log

☐ Std Error File

Using this feature provides you with two major benefits.

1. If you noticed, enabling this feature also provides you with the ability to indicate the log file retention period. In the example above, ActiveBatch will delete the log file after one (1) day. That's one less thing you need to do.
2. Your log file can now be viewed from any participating machine without you having to first copy it to your local system. The ability to view a log file can be found on the Job History tab.



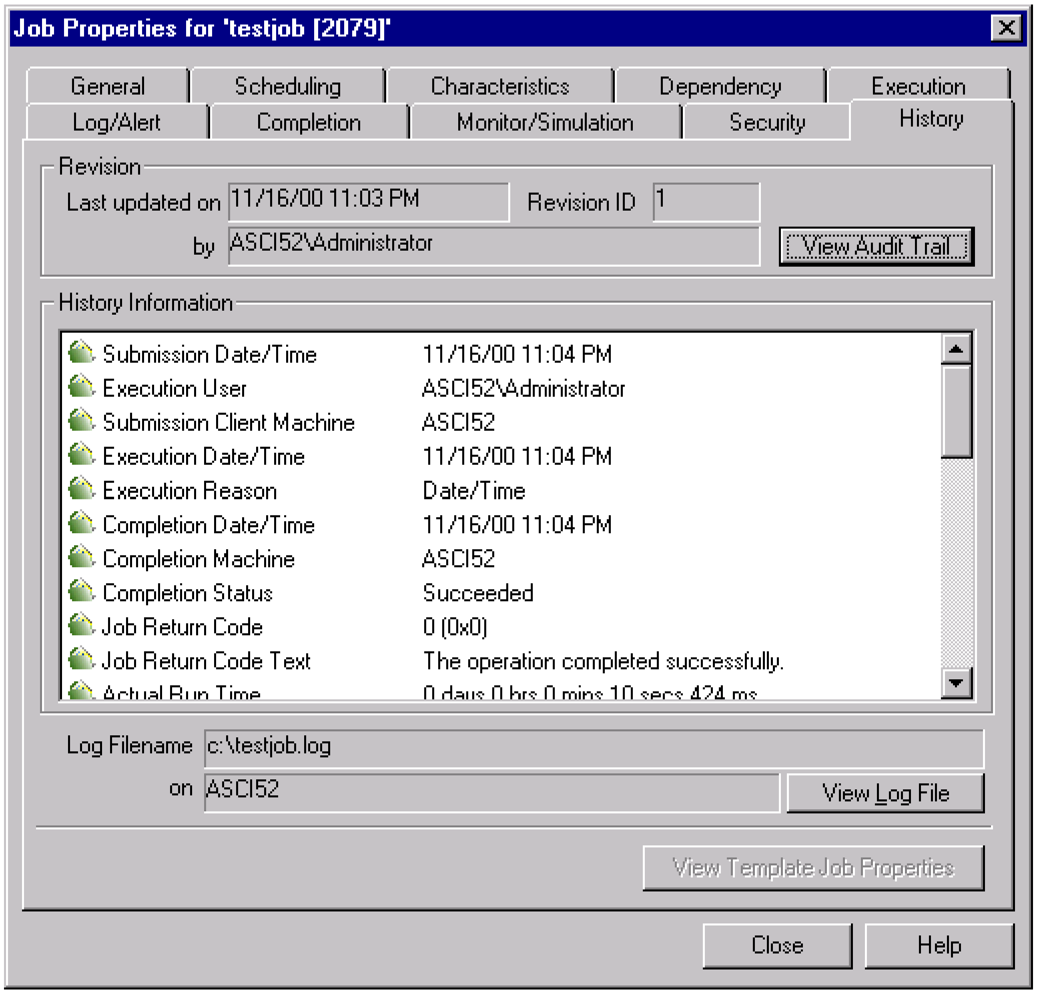
Log Filename \\ASCI52\ASCI_ABATLOG\ASCI52\testq\job-a_002078_16Nov2000-170521.LO

on ASCI52

Clicking the *View Log File* button will cause an editor to be started with the job's log file. By default, Notepad is the editor used. You can change the default editor via the Tools/Options menu.

7.4.2 Job History

The Job History tab is created as soon as a job is submitted or an instance created. Normal or Job instances appear like this:



Revision

Last updated on	11/16/00 11:03 PM
Revision ID	1
by	ASCI52\Administrator

History Information

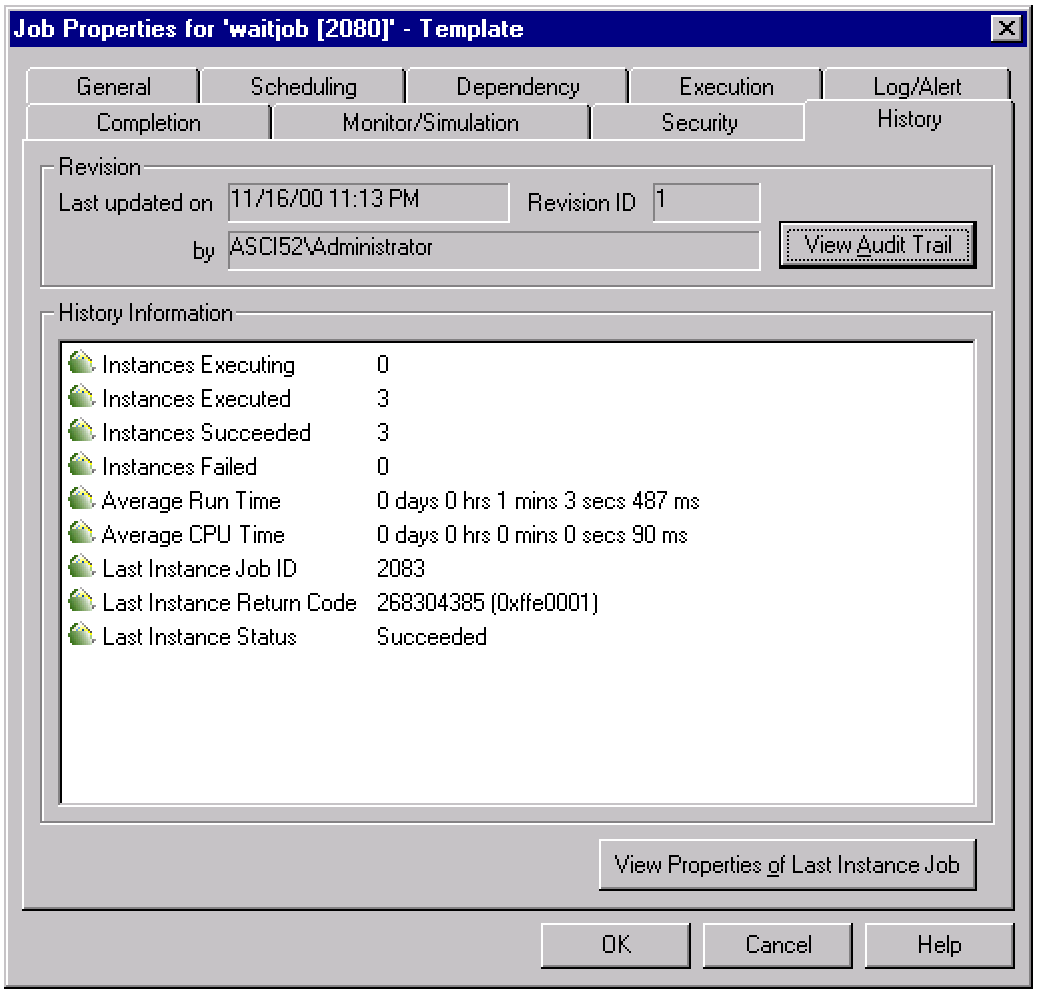
Submission Date/Time	11/16/00 11:04 PM
Execution User	ASCI52\Administrator
Submission Client Machine	ASCI52
Execution Date/Time	11/16/00 11:04 PM
Execution Reason	Date/Time
Completion Date/Time	11/16/00 11:04 PM
Completion Machine	ASCI52
Completion Status	Succeeded
Job Return Code	0 (0x0)
Job Return Code Text	The operation completed successfully.
Actual Run Time	0 days 0 hrs 0 mins 10 secs 424 ms

Log Filename c:\testjob.log

on ASCI52

Note the Revision History information area.

Template Jobs history tab appears like this:



Revision

Last updated on 11/16/00 11:13 PM Revision ID 1

by ASCI52\Administrator

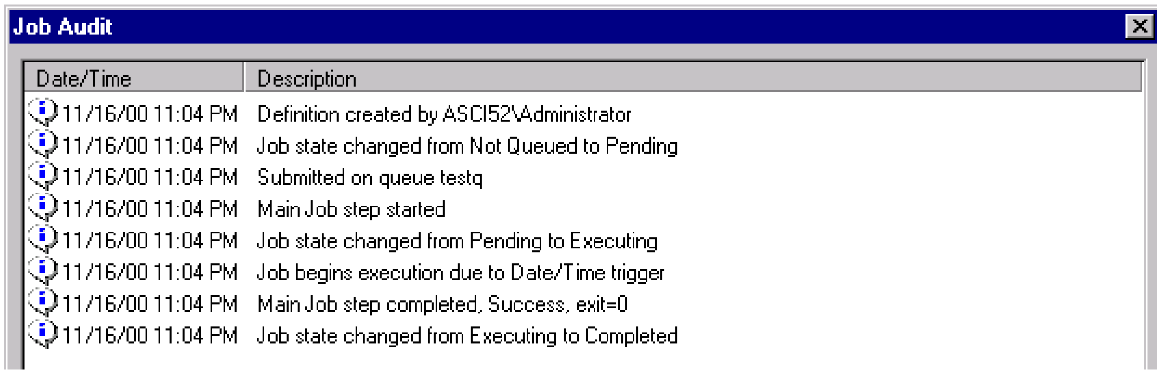
History Information

Instances Executing	0
Instances Executed	3
Instances Succeeded	3
Instances Failed	0
Average Run Time	0 days 0 hrs 1 mins 3 secs 487 ms
Average CPU Time	0 days 0 hrs 0 mins 0 secs 90 ms
Last Instance Job ID	2083
Last Instance Return Code	268304385 (0xffe0001)
Last Instance Status	Succeeded

From the Job History tab you can retrieve Job Audits as well as view your log file. If the Job is a template you can also retrieve the properties of the last instance. If the Job is an instance you can retrieve the Job's template. Retrieval of these properties are nested in nature.

7.4.2.1 Audits

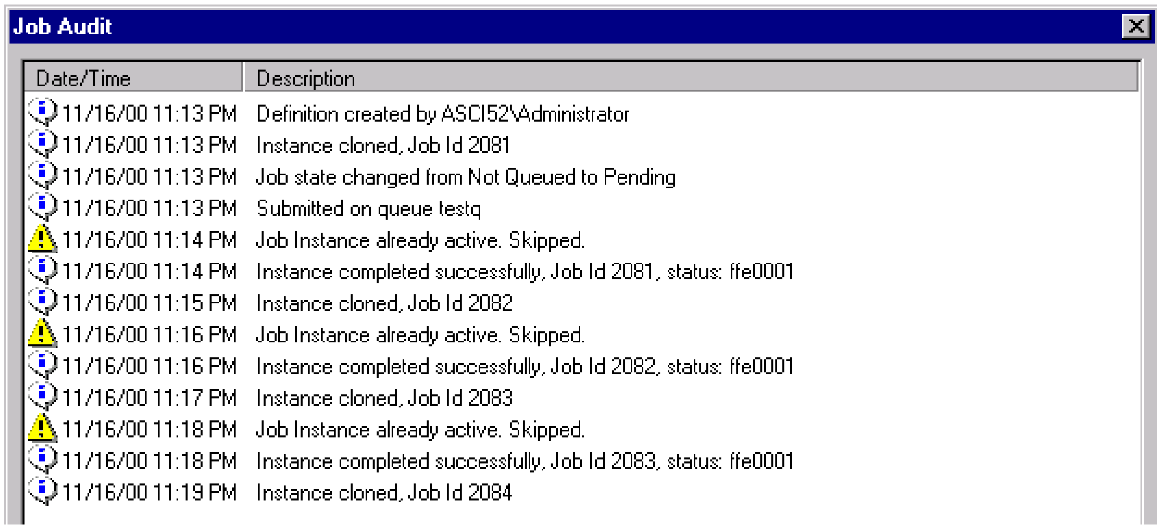
The Job Audit area is the first place to look when a question arises as to ActiveBatch/Job interaction. All exchanges between ActiveBatch and the Job are recorded in this area.



Job Audit

Date/Time	Description
11/16/00 11:04 PM	Definition created by ASCI52\Administrator
11/16/00 11:04 PM	Job state changed from Not Queued to Pending
11/16/00 11:04 PM	Submitted on queue testq
11/16/00 11:04 PM	Main Job step started
11/16/00 11:04 PM	Job state changed from Pending to Executing
11/16/00 11:04 PM	Job begins execution due to Date/Time trigger
11/16/00 11:04 PM	Main Job step completed, Success, exit=0
11/16/00 11:04 PM	Job state changed from Executing to Completed

The audit box extract above is for a normal job and tracks the job from submission to end.



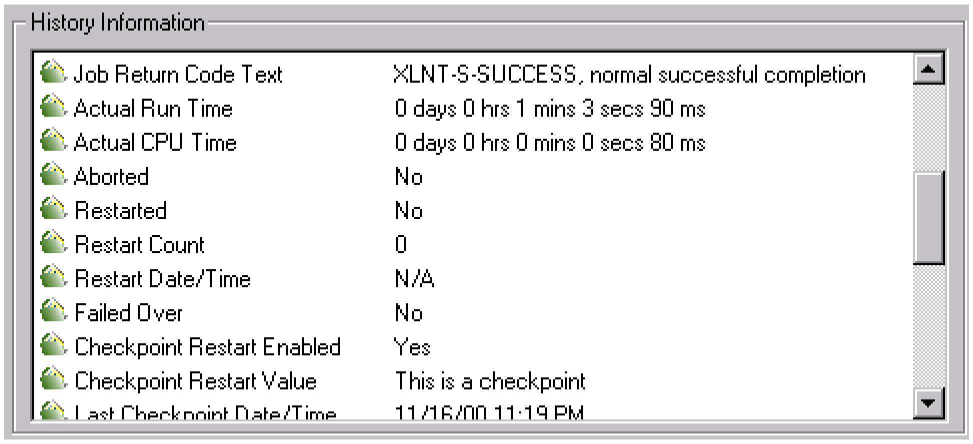
Job Audit

Date/Time	Description
11/16/00 11:13 PM	Definition created by ASCI52\Administrator
11/16/00 11:13 PM	Instance cloned, Job Id 2081
11/16/00 11:13 PM	Job state changed from Not Queued to Pending
11/16/00 11:13 PM	Submitted on queue testq
11/16/00 11:14 PM	Job Instance already active. Skipped.
11/16/00 11:14 PM	Instance completed successfully, Job Id 2081, status: ffe0001
11/16/00 11:15 PM	Instance cloned, Job Id 2082
11/16/00 11:16 PM	Job Instance already active. Skipped.
11/16/00 11:16 PM	Instance completed successfully, Job Id 2082, status: ffe0001
11/16/00 11:17 PM	Instance cloned, Job Id 2083
11/16/00 11:18 PM	Job Instance already active. Skipped.
11/16/00 11:18 PM	Instance completed successfully, Job Id 2083, status: ffe0001
11/16/00 11:19 PM	Instance cloned, Job Id 2084

The audit box extract above is for a template job. Note the message for each instance cloned and each instance that completes. The yellow warning symbol notes that instances were skipped because a preceding instance was active and the job has "On Job Active Skip" enabled.

7.4.2.2 History

This section describes a few of the job history properties. For a complete explanation of all fields, see Section 6.5.3.10.



History Information

Job Return Code Text	XLNT-S-SUCCESS, normal successful completion
Actual Run Time	0 days 0 hrs 1 mins 3 secs 90 ms
Actual CPU Time	0 days 0 hrs 0 mins 0 secs 80 ms
Aborted	No
Restarted	No
Restart Count	0
Restart Date/Time	N/A
Failed Over	No
Checkpoint Restart Enabled	Yes
Checkpoint Restart Value	This is a checkpoint
Last Checkpoint Date/Time	11/16/00 11:19 PM

Note the actual run time (elapsed) and CPU time used. This sample job took a checkpoint that is also noted in the history. The history information displayed above is only a portion of that which is available.

7.5 Security

Each ActiveBatch object is securable. The Windows 2000/NT model is used. ActiveBatch Administrators have the ability to establish default security for each object. Earlier sections described the actual access permissions that can be granted to each object. This section assumes the reader is familiar with this model and provides a more advanced discussion on security. ActiveBatch security operations can be performed through most Client interfaces (except on Windows 9x/Me systems). Default object security can be scripted and/or accessed through the ActiveBatch MMC snap-in.

ActiveBatch supports *Grant* and *Deny* access control entries (ACE). Using the NT convention, Deny and then Grant ACEs appear. This means that it's pretty simple to deny an account or group. ActiveBatch goes one step further in that individual operations can be denied. Remember that NT is a positive authorization system and that unless you (or a group) are authorized you won't be able to gain access to the object. If you want to deny an account access that would otherwise be granted under a group membership, simply specify a Grant Noaccess ACE to that account or group.

The owner of an ActiveBatch object is granted Full Control access. This access cannot be modified or removed. If you attempt to remove all ACEs from an object you will receive a warning that no one but the owner of the object will be able to perform operations. ActiveBatch Administrator users can always Take Ownership of an object.

8 ADMINISTERING ACTIVEBATCH THROUGH MMC

This section concerns use of the Microsoft Management Console (MMC) which is software specific to Microsoft Windows 2000/NT systems.

ActiveBatch provides integrated support of Microsoft's Management Console (MMC). The goal of the MMC is to minimize the cost of administering Windows-based environments, and to provide a simple, consistent, and integrated administration user-interface model. The MMC itself does not administer or manage anything. Instead it provides a framework through which a varying array of different system and application products can achieve a common and consistent interface. Each application contributes a *Snap-In* to the MMC. A Snap-in can be though of as an execution instance of the MMC. Snap-ins actually do all the work. ActiveBatch's support of the MMC is through an ActiveBatch Snap-in. ActiveBatch requires a minimum of MMC Version 1.2.

When you install ActiveBatch, the installation process examines your machine and determines whether you have the MMC core components installed. MMC is optional for Windows NT V4 (MMC can be installed through a Microsoft Option Pack or using the ActiveBatch product CD) and is integrated into Windows 2000. ActiveBatch will install and register the snap-in regardless of whether MMC is actually present.

8.1 Adding the Snap-In

8.1.1 Using the ActiveBatch MSC Snap-In

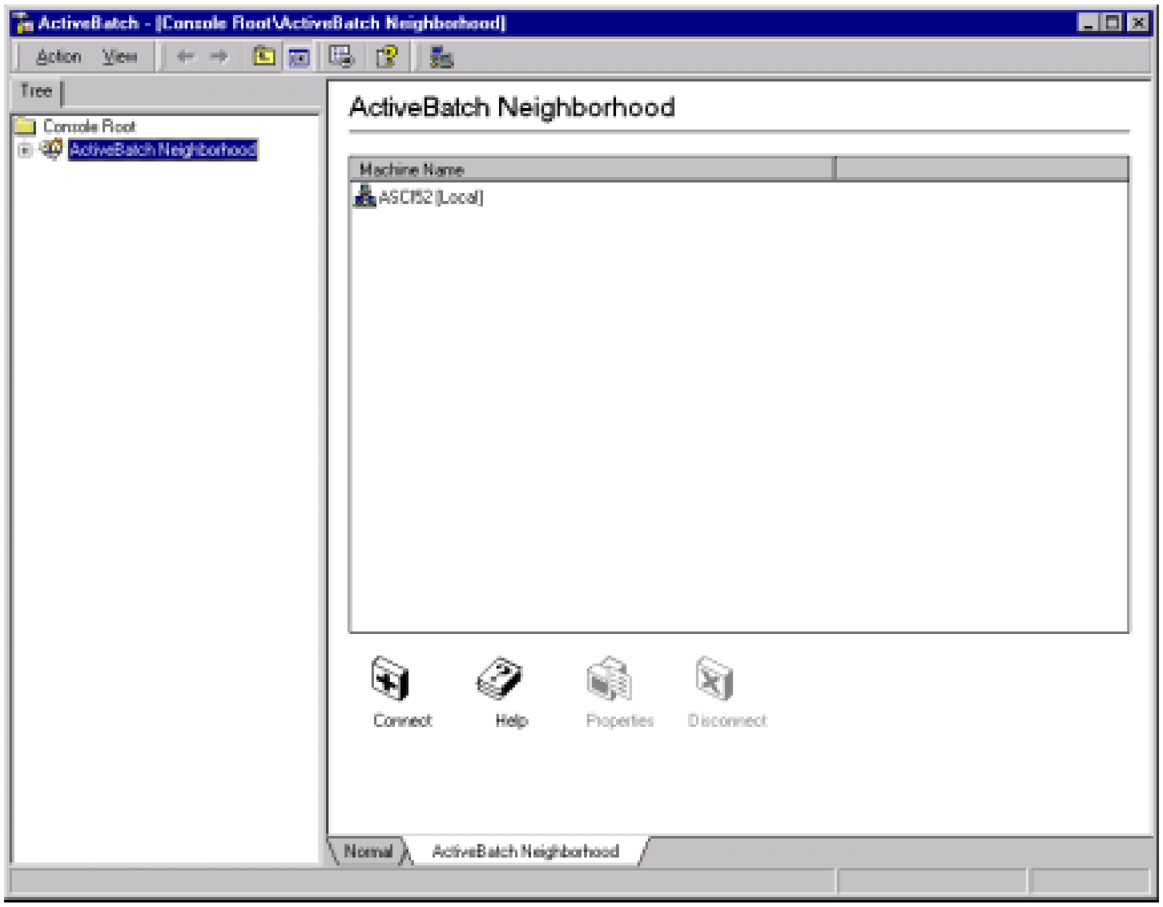
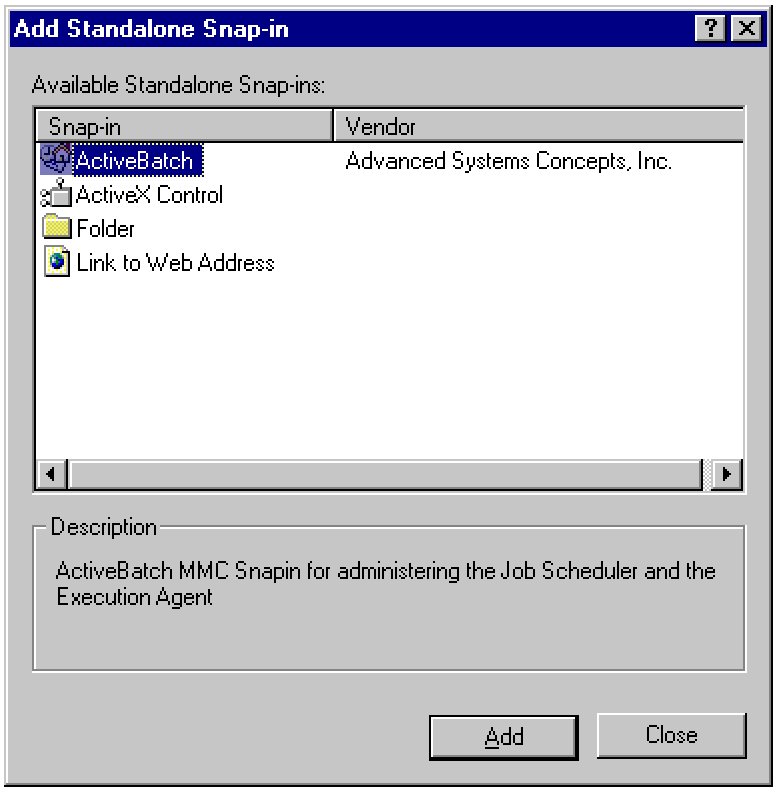


Figure 121 - MMC Console

The ActiveBatch product installation provides a file named *ActiveBatch.MSC* (located in the ActiveBatch Installation directory and in the ActiveBatch Program Folder) which contains a simple to use MMC console interface. Figure 121 illustrates the console interface.

8.1.2 Manually Adding the Snap-In

To manually add the ActiveBatch snap-in, select the *Console* menu on the menu bar and click on *Add/Remove Snap-in…*. When the Add/Remove Snap-in dialog box appears click the *Add* button. Figure 122 will appear listing ActiveBatch as one of the snap-ins you may add. Select ActiveBatch and click *Add*.

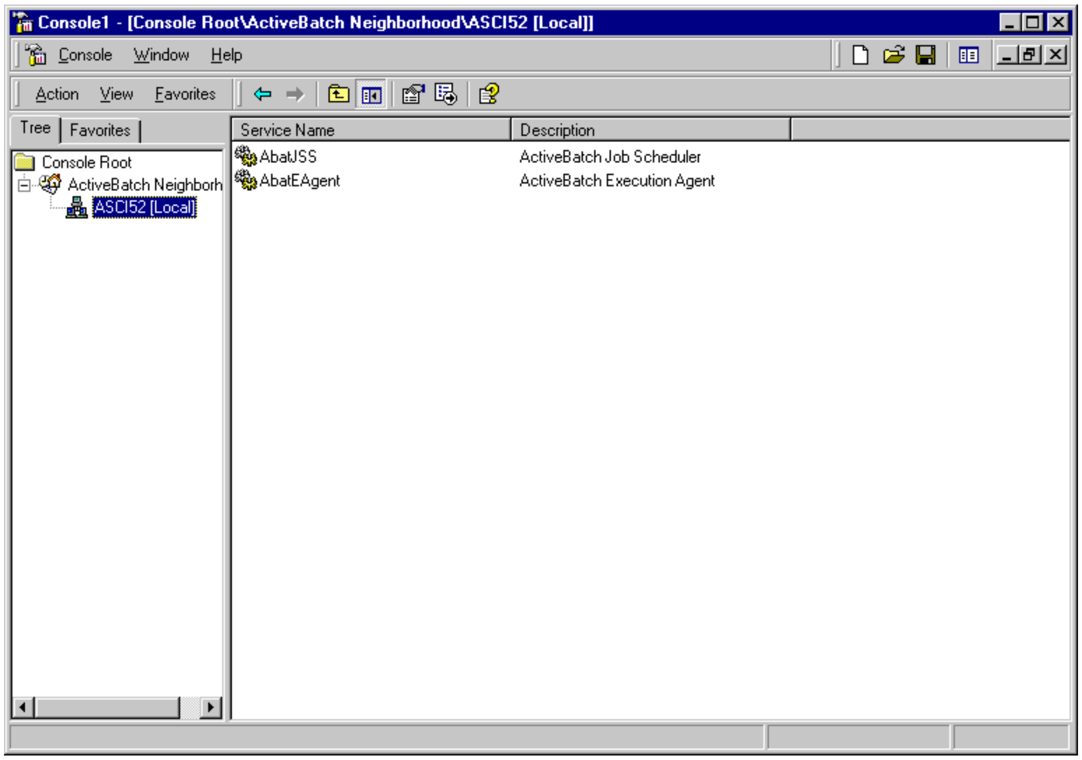


Snap-in	Vendor
ActiveBatch	Advanced Systems Concepts, Inc.
ActiveX Control	
Folder	
Link to Web Address	

ActiveBatch MMC Snapin for administering the Job Scheduler and the Execution Agent

Figure 122 - Add ActiveBatch Snap-in

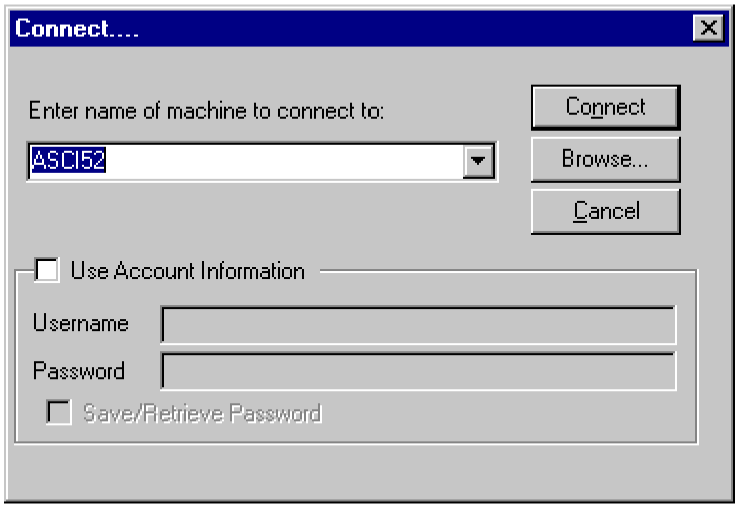
Once you have closed this dialog and then clicked *OK* to confirm the snap-in, the initial screen will change to one that approximates the following.



Service Name	Description
AbatJSS	ActiveBatch Job Scheduler
AbatEAgent	ActiveBatch Execution Agent

Figure 123 - MMC ActiveBatch Neighborhood

By default, your local machine will appear as the first computer under the ActiveBatch Neighborhood root.



Enter name of machine to connect to:
ASCI52

☐ Use Account Information

Username

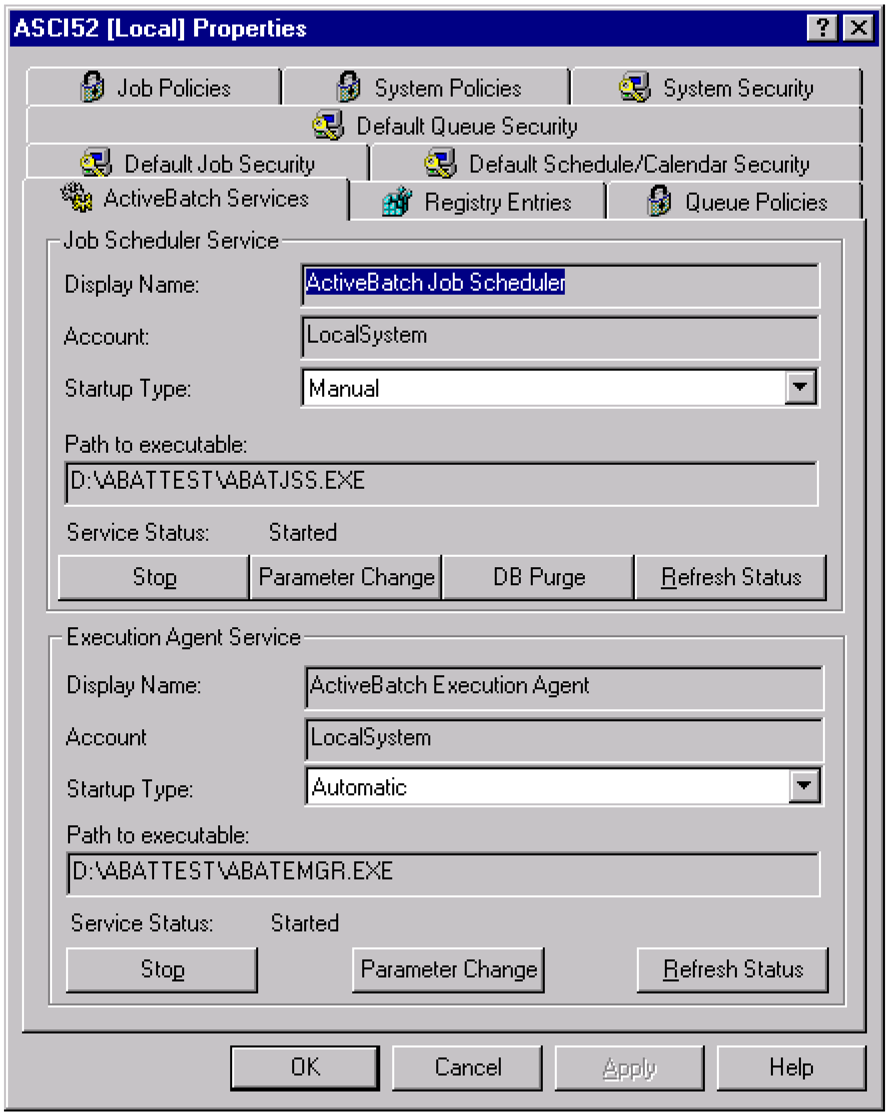
Password

☐ Save/Retrieve Password

Figure 124 - MMC Connect Dialog

To connect to other ActiveBatch systems, simply right-click on the ActiveBatch Neighborhood item, select Connect and specify the machine you want to connect to. Figure 124 will appear requesting your security credentials.

To access the various administrative and management aspects of ActiveBatch, right-click on the machine you want to examine or manage and select *Properties*.


ASCI52 [Local] Properties

Job Scheduler Service

Display Name: ActiveBatch Job Scheduler

Account: LocalSystem

Startup Type: Manual

Path to executable:
D:\ABATTEST\ABATJSS.EXE

Service Status: Started

Execution Agent Service

Display Name: ActiveBatch Execution Agent

Account: LocalSystem

Startup Type: Automatic

Path to executable:
D:\ABATTEST\ABATEMGR.EXE

Service Status: Started

Figure 125 - MMC Properties

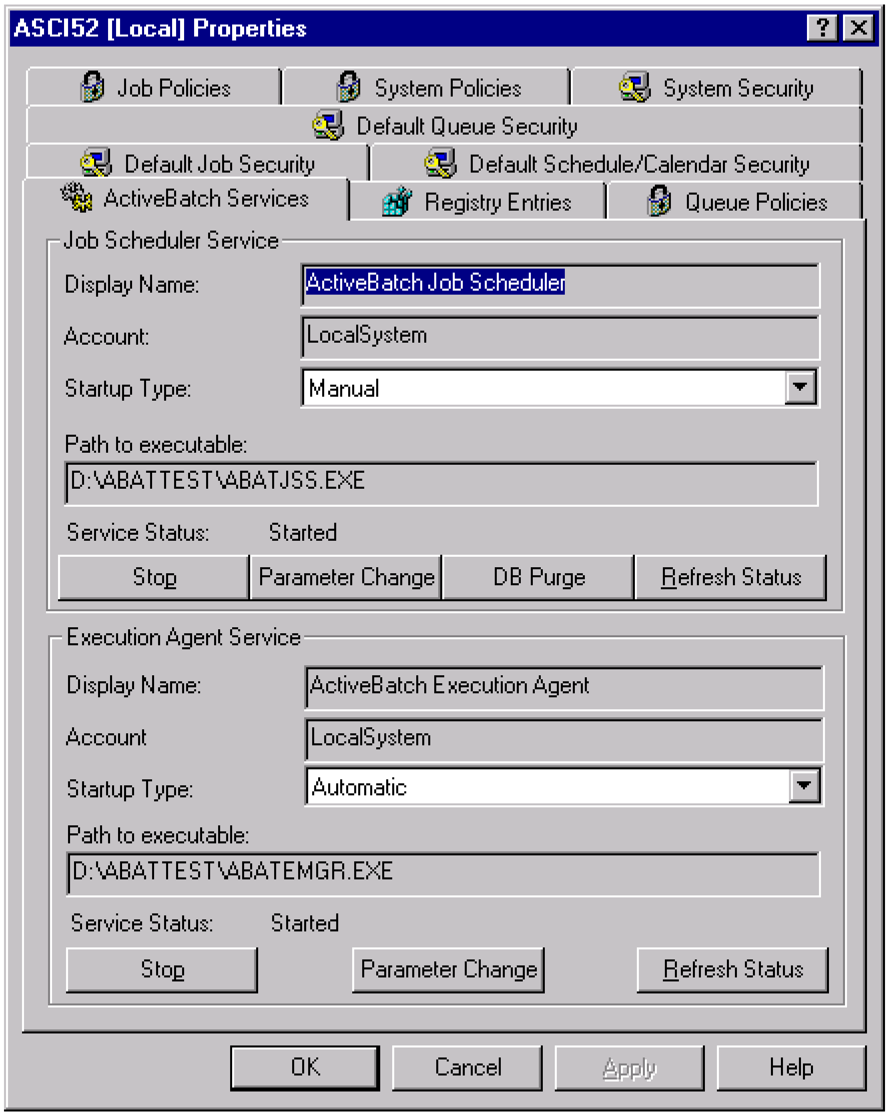
The Properties screen contains nine (9) tab sheets: ActiveBatch Services, Registry Entries, Queue Policies, Job Policies, System Policies, Default Queue, Job, Schedule/Calendar Security and System Security. Remember that you must click *OK* or *Apply* to confirm and apply your selections. The next sections will more fully describe each of these tab sheets.

8.2 ActiveBatch Services

This section deals with starting and stopping the ActiveBatch Services for Windows 2000/NT.

Two (2) services are typically used with ActiveBatch: Job Scheduler and Execution Agent. Depending on the Product Edition you bought and how you decided to distribute the software, one or more of these services may be running (or need to run). The ActiveBatch Snap-in provides the ability to start and stop these services on a local or remote machine. Every ActiveBatch system requires at least one (1) Job Scheduler. You can run multiple Job Schedulers on multiple machines if your license allows that. Similarly you can run one or more Execution Agents on one or more machines if your license allows that. The advantages of running more than one Job Scheduler are:

1. Higher Availability. A Job Scheduler machine outage won't affect your entire Enterprise.
2. Scalability. Some high-end users may need to schedule thousands of complex jobs. The ability to add additional Job Scheduler machines means that ActiveBatch scheduling capability can also scale.
3. Organizational/Application Divisions. Geography and/or application requirements may necessitate the need for multiple concurrent Job Schedulers.


ASCI52 [Local] Properties

Job Scheduler Service

Display Name: ActiveBatch Job Scheduler

Account: LocalSystem

Startup Type: Manual

Path to executable:
D:\ABATTEST\ABATJSS.EXE

Service Status: Started

Execution Agent Service

Display Name: ActiveBatch Execution Agent

Account: LocalSystem

Startup Type: Automatic

Path to executable:
D:\ABATTEST\ABATEMGR.EXE

Service Status: Started

Figure 126 - MMC ActiveBatch Services

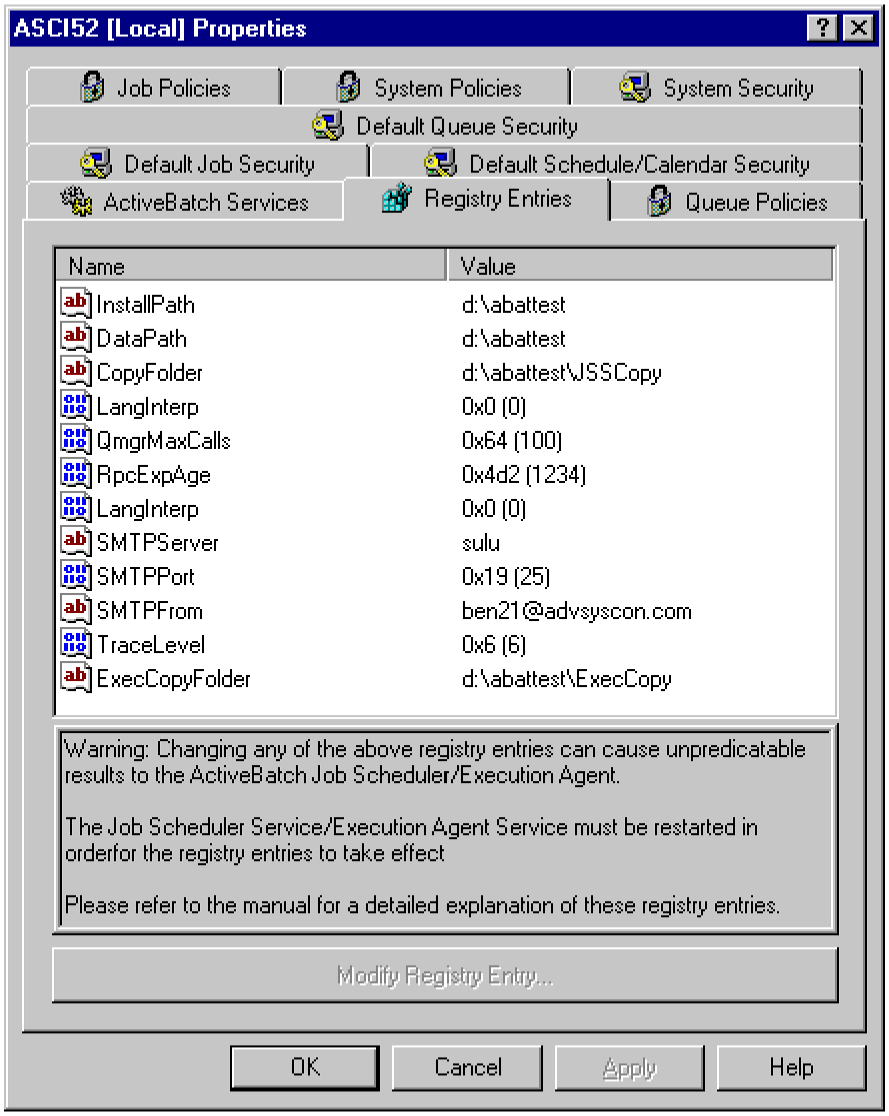
This tab sheet controls the ActiveBatch Job Scheduler and/or Execution Agent services. The only requirement to controlling these services is that they first be installed on the connected machine. You may stop or start each service as well as change the startup mode of the service. If the service section is grayed out the service is not installed on the selected machine. The *Service*

Status field indicates the current service status (usually Stopped or Started). The *Startup Type* is a drop down allowing you to select either *Automatic*, *Manual* or *Disabled*. The Start or Stop button displays the correct operation to perform depending on the service's state. To refresh the service's status, click the *Refresh Status* button. The button *Parameter Change* causes the selected service to re-read those registry settings which can be changed on a dynamic basis. See Section 3.2.1 for more information. For the Job Scheduler service the *DB Purge* button allows you to run a Database Purge immediately.

8.3 ActiveBatch Registry Entries

This section allows you to change various registry entries that pertain to ActiveBatch operation.

To change one of the entries below simply select the entry and either double-click or click the *Modify Registry Entry* button. A dialog box will appear allowing you to change the entry. Some entries cannot be changed (in particular InstallPath cannot be changed). Also some entries are not displayed (for example, those pertaining to Serial Numbers aren't displayed and can't be changed through the MMC).



Name	Value
InstallPath	d:\abattest
DataPath	d:\abattest
CopyFolder	d:\abattest\JSSCopy
LangInterp	0x0 (0)
QmgrMaxCalls	0x64 (100)
RpcExpAge	0x4d2 (1234)
LangInterp	0x0 (0)
SMTPServer	sulu
SMTPPort	0x19 (25)
SMTPFrom	ben21@advsyscon.com
TraceLevel	0x6 (6)
ExecCopyFolder	d:\abattest\ExecCopy

Warning: Changing any of the above registry entries can cause unpredicatable results to the ActiveBatch Job Scheduler/Execution Agent.

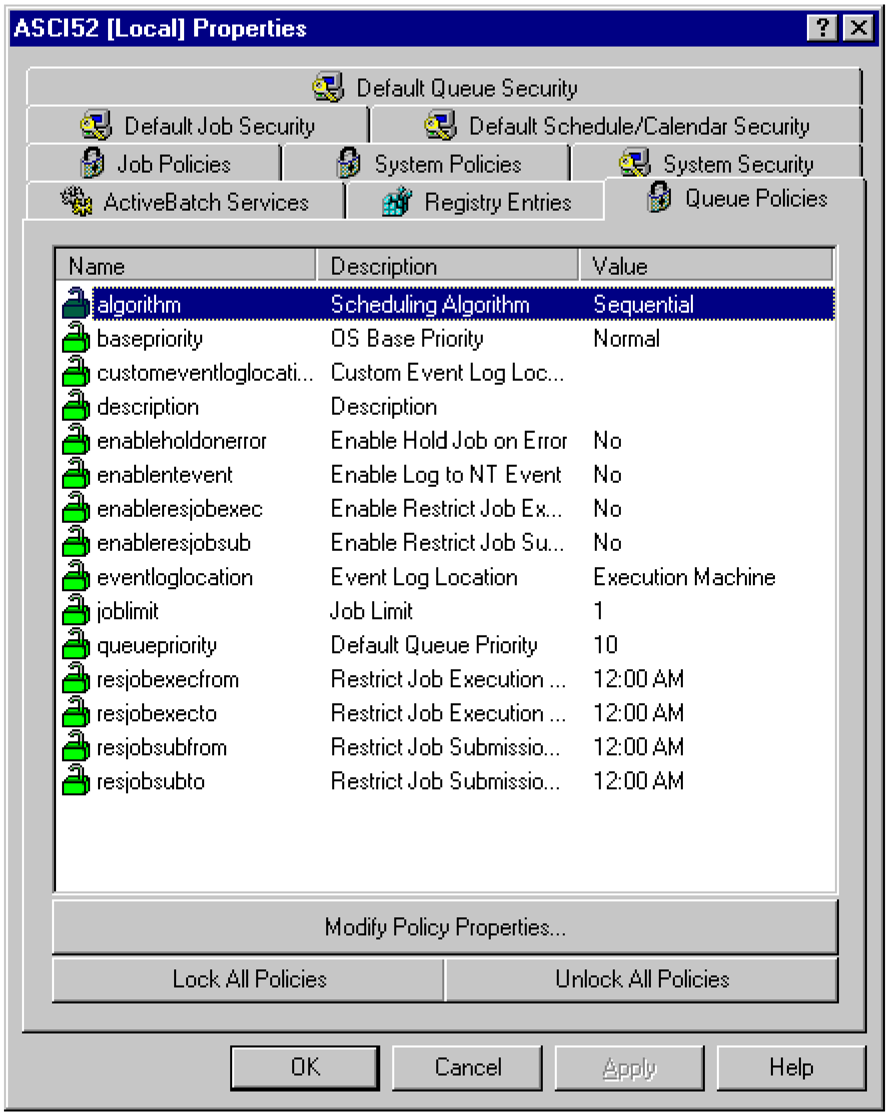
The Job Scheduler Service/Execution Agent Service must be restarted in orderfor the registry entries to take effect

Please refer to the manual for a detailed explanation of these registry entries.

Figure 127 - MMC Registry Entries

Information on these entries can be found in Section 3.2.1.

8.4 ActiveBatch Policy Management



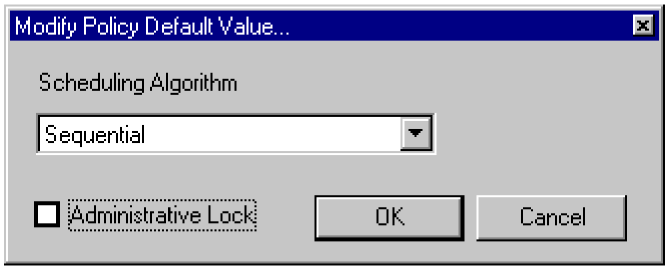
Name	Description	Value
algorithm	Scheduling Algorithm	Sequential
basepriority	OS Base Priority	Normal
customeventloglocati...	Custom Event Log Loc...	
description	Description	
enableholdonerror	Enable Hold Job on Error	No
enablentevent	Enable Log to NT Event	No
enableresjobexec	Enable Restrict Job Ex...	No
enableresjobsub	Enable Restrict Job Su...	No
eventloglocation	Event Log Location	Execution Machine
joblimit	Job Limit	1
queuepriority	Default Queue Priority	10
resjobexecfrom	Restrict Job Execution ...	12:00 AM
resjobexecto	Restrict Job Execution ...	12:00 AM
resjobsubfrom	Restrict Job Submissio...	12:00 AM
resjobsubto	Restrict Job Submissio...	12:00 AM

Figure 128 - MMC Policies

This section deals with the management of various ActiveBatch policies.

ActiveBatch provides Administrators with the ability to set ActiveBatch *Policies*. A Policy is a default value that may be set and locked for use by ActiveBatch for the various objects it supports. Most policies concern object property values however several concern the overall system.

All policies have two (2) characteristics. One, the default value that should be used for an object's or system's property and, two, whether the default value should be "locked" and made non-modifiable for non-Administrative users. If a policy is "locked" then the value cannot be changed unless the user is an ActiveBatch Administrator. This allows an administrator to set values that cannot be overridden unless the user is sufficiently privileged. Figure 128 shows the policies that affect a Queue. To modify a policy, simply select the policy and then click the large *Modify Policy Properties* button.


Modify Policy Default Value...

Scheduling Algorithm: Sequential

☐ Administrative Lock

Figure 129 - MMC Modify Policy

The figure above illustrates the modification of the Queue Scheduling Algorithm policy. You may change the default value using the drop down and/or you may select the *Administrative Lock* checkbox to prevent non-ActiveBatch Administrators from changing the default value. Click *OK* to confirm and accept the selection.

ActiveBatch provides three (3) types of policies: System, Queue and Job.

8.4.1 System Policies

The table below describes the System policies that may be manipulated by an ActiveBatch Administrator through the MMC snap-in. You may view these policies through ABATAdmin. Please note that you cannot lock any of the System Policies.

Name	**Default Value**	**Description**
CopyScriptFileMaxSize	20000	This value (in bytes) represents the maximum file length of any pre/post or main job step that can be copied by ActiveBatch.
EnableDBPurgeRun	Yes	Should Job Scheduler auto run DBPurge?
DBPurgeAuditTemplate RetentionPeriod	7	The period of time (in days) in which audits for a template job are removed beginning with the oldest audits.
DBPurgeTime	12:00AM	Time that DBPurge should run if DBPurgeRun is enabled.
EnableDBPurgeLog	Yes	Should DBPurge log its activities? If Yes, the file *DBPurge.log* is created in the ActiveBatch Installation directory.
LogNTEventOnJobDelete	No	Should DBPurge log to the NT Event Log when it deletes a job?
JobDelayStartMax	5	Amount of time (in minutes) from when a job is eligible to start and when it actually executes that the job should be considered "delayed" from an alert point-of-view.

Table 10 - System Policies

8.4.2 Queue Policies

The table below describes the Queue policies that may be manipulated by an ActiveBatch Administrator through the MMC snap-in. You may view these policies through ABATAdmin. These policies are used by all client and COM interfaces when a new Queue is created. See Section 6.3 for more specific information concerning these Queue properties. Any queue policy may be "locked" to prevent the policy value from being changed by a non-ActiveBatch Administrator.

Name	Default Value	Description
Algorithm	Sequential	Scheduling Algorithm for Generic Queues. Sequential, Round Robin and Workload Balancing are possible choices.
Basepriority	Normal	OS Base Priority. Any Windows 2000/NT priority or a number from 0-31.
Description		Queue Description.
EnableNTEvent	No	Enable NT Event Log?
EnableHoldOnError	No	Enable Hold on Error?
EnableResJobExec	No	Enable Restrict Job Execution Times?
EnableResJobSub	No	Enable Restrict Job Submission Times?
EventLogLocation		NT Event Location
JobLimit	1	Job Limit (Range: 1-100)
QueuePriority	10	Queue Priority (Range: 1-100)
ResJobExecFrom	12:00AM	Restrict Job from Executing *From* time
ResJobExecTo	12:00AM	Restrict Job from Executing *To* time
ResJobSubFrom	12:00AM	Restrict Job from Submission *From* time
ResJobSubTo	12:00AM	Restrict Job from Submission *To* time

Table 11 - Queue Policies

8.4.3 Job Policies

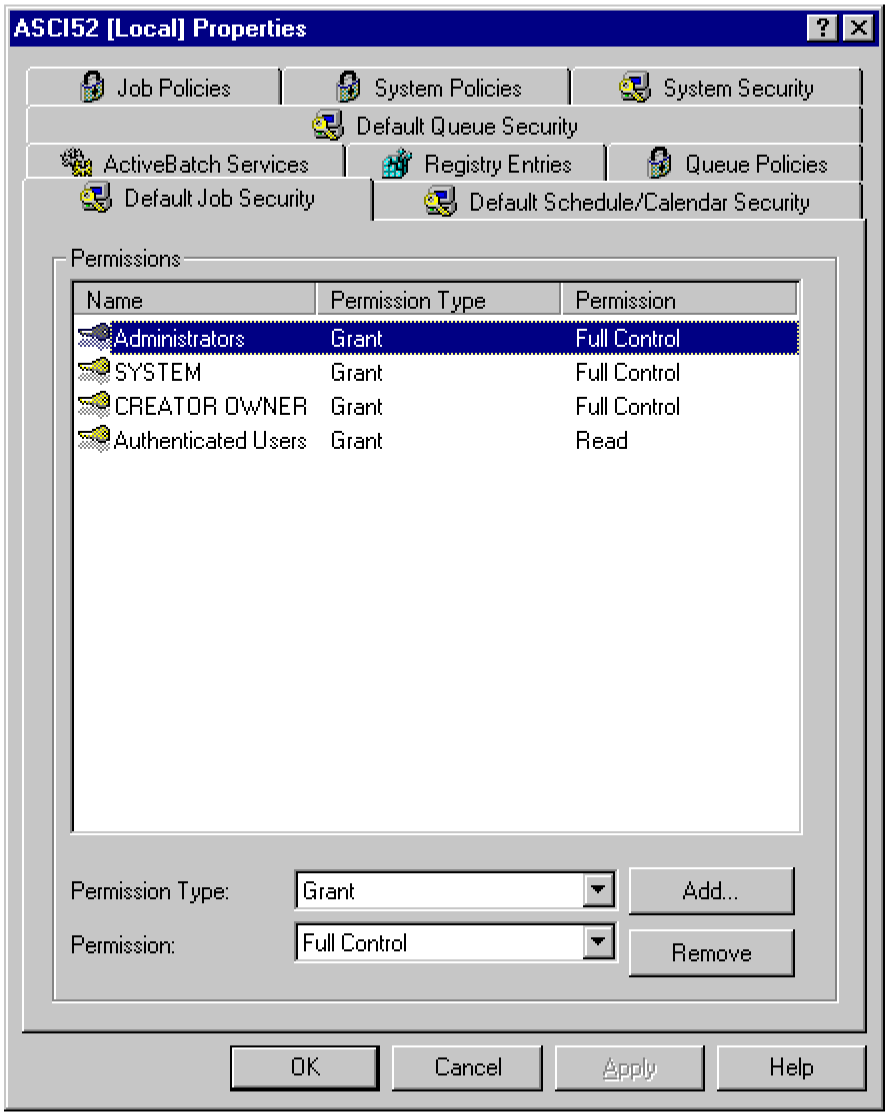
The table below describes the Job policies that may be manipulated by an ActiveBatch Administrator through the MMC snap-in. You may view these policies through ABATAdmin. These policies are used by all client and COM interfaces when a new Job is created. See Section 6.5.2 for more specific information concerning these Job properties. Any job policy may be "locked" to prevent the policy value from being changed by a non-ActiveBatch Administrator.

Name	Default Value	Description
CopyFile	No	Copy Main file to execution machine?
CopyPre	No	Copy Pre Job Step file to execution machine?
CopyPost	No	Copy Post Job Step file to execution machine?
Description		Job Description.
EnableABatTrigger	No	Enable ActiveBatch Triggers?
EnableCentralLog	No	Enable Central Log File Management?
EnableDateTime	Yes	Enable Date/Time Triggers?
EnableEventrigger	No	Enable Event Triggers?
EnableHoldJobComp	No	Enable Hold Job on Completion?
EnableHoldJobOnError	No	Enable Hold Job on Error?
EnableHoldJobOnSubmit	No	Enable Hold Job on Submission?
EnableLogging	Yes	Enable Job Logging?
EnableProcMask	No	Enable Processor Mask? (SMP)
EnablePostFail	No	Enable Post-Step fails then job fails?
EnablePostFile	No	Enable Post-Step file?
EnablePreFile	No	Enable Pre-Step file?
EnableReadOnly	No	Enable Job Read Only?
EnableRetainLog	Yes	Enable Log File Retention?

Name	Default Value	Description
EnableSaveHistory	Yes	Enable Save Job History?
ProcMask	All	SMP Processor Affinity Bit-Mask (ALL means all processors used)
MaxRestarts	0	Maximum Number of Restarts (0=Unlimited)
OnActive	Skip	If Job Active then… Skip, Wait, Run Multiple
OnCrashFailover	No	If Machine crashes, enable failover
OnCrashWait	No	If Machine crashes, wait for machine to reboot
OnProcFail	No	On Process Failure
Qselection	Any	Generic Queue Selection… ANY, ALL
RetainLogFor	1	Retain Log Files (if centrally managed) for *n* days
RunOnce	No	Run Job Once regardless of triggers
SaveHistory	5	If enabled, Save Job History for *n* days
SuccessCodeRule	0	Success Code Rules. *n, n-x, ODD, EVEN, LZERO, NTMSG.*
TerminateAllChildProc	No	Terminate all Child Processes when parent completes.
TimeZone	Job Scheduler	Time Zone to use for scheduling. Job Scheduler, Client, UTC

Table 12 - Job Policies

8.5 Assigning Default Object Security



Name	Permission Type	Permission
Administrators	Grant	Full Control
SYSTEM	Grant	Full Control
CREATOR OWNER	Grant	Full Control
Authenticated Users	Grant	Read

Permission Type: Grant

Permission: Full Control

Figure 130 - MMC Default Job Security

Just as with other Windows objects you can assign default security access that is used whenever an ActiveBatch object is created. The MMC Snap-in allows you to control default security access for new Queues, Jobs, Schedules and Calendars. To add, remove or change security access click on the object's default security tab. A discussion of the actual security access mask for each object can be found in the section on each object's security.

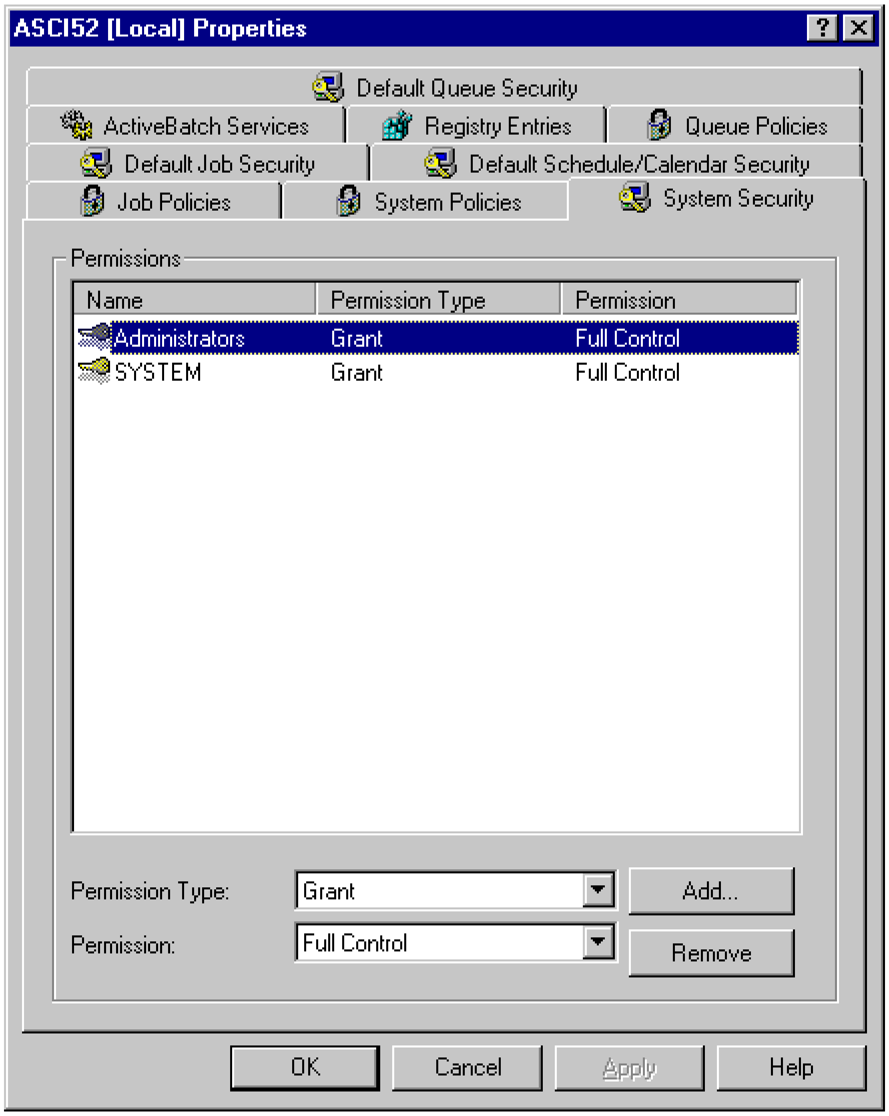
The figure above lists the default accounts and access that is to be applied to a new Job object. To add a new account, click the *Add* button. To edit or remove a account/access, select the account and either change the access or click the *Remove* button.

8.6 ActiveBatch Administrators

ActiveBatch Administrator status is important to the product. An ActiveBatch Administrator can modify policies, create queues and, by default, access all ActiveBatch objects. While most customers will elect to have Windows Administrators also manage their ActiveBatch systems the product is flexible to allow further delegation. This means that you can decide to combine or keep the functions separate. By default, the ActiveBatch installation process adds the *Administrators* group as a member. This allows NT administrators the ability to, at least initially, manage the product "out-of-the-box".

ActiveBatch allows you to further delegate who can create new Queues. Sometimes it may be important to allow an ActiveBatch non-Administrator to create a new queue.

This functionality can be managed using the System Security tab.



Name	Permission Type	Permission
Administrators	Grant	Full Control
SYSTEM	Grant	Full Control

Permission Type: Grant

Permission: Full Control

Figure 131 - MMC System Security

The figure above lists two accounts that are considered ActiveBatch Administrators. The actual permission that grants such status is labeled *Executive*. *Manage* access allows a trustee to create a new queue. *Full Control* provides the ability for the account to do both.

9 Using the ActiveBatch Command-Line Interface

Most of this manual concerns the use of ActiveBatch with either the ABATAdmin program or the ActiveBatch MMC Snap-in. This section describes the set of commands available through the console interface. Console commands are useful in that you can always issue these commands regardless of scripting language capability. The console command interface doesn't provide access to the entire set of ActiveBatch capability. For programmatic access you are encouraged to read Section 0 on "ActiveBatch Scripting and Programming" and the use of Microsoft COM automation with ActiveBatch.

The commands that follow are divided into two categories: Queue and Job. Queue commands create and otherwise manage the batch queue system itself. Job commands are those necessary for submitting a job to a queue for background execution and also for managing those jobs within the batch system.

The commands that follow require the verb ABAT to execute in a CMD console. If you have XLNT (Version 3 or later) installed you don't need to preface the command. For example:

```
ABAT SUBMIT job-name /QUEUE=test-queue
```

This command would be invoked within the CMD console subsystem. If you are using ActiveBatch with XLNT you can invoke the commands below just as you would other XLNT commands. For example, the same command above would be:

```
$ SUBMIT job-name /QUEUE=test-queue
```

Note: The /ON qualifier can be used to direct requests to any licensed and active ActiveBatch Job Scheduler. The /ON qualifier also supports the specification of TCP/IP fully qualified hostnames as well as IP addresses.

The command qualifiers (with some minor exceptions) use the Queue and Job Policies to establish defaults.

9.1 Conventions

Certain commands and qualifiers accept date and time values. You can specify these values in one of the following formats:

9.1.1 Absolute Time

An absolute time is a specific date or time of day. The format for an absolute time is as follows:

"[dd-mmm-yyyy][:hh:mm:ss]"

The fields are:

dd	Day of month (range from 1 to 31, depending on month)
mmm	Month (JAN...DEC)
yyyy	Year
hh	Hour (range from 0 to 23)
mm	Minute (range from 0 to 59)
ss	Second (range from 0 to 59)

The following rules apply to an absolute time:

- You can truncate the date or the time from the right.
- The date must contain at least one hyphen.
- You can omit any of the fields (other than month if a date is specified) as long as you include the punctuation marks that separate the fields.
- A truncated or omitted date field defaults to the corresponding fields for the current date.
- A truncated or omitted time field defaults to zero.

Example:

1-DEC-1997	Midnight, December 1, 1997.
15-DEC-1997:13:00	1:00 PM, December 15, 1997
20-DEC::30	12:30 AM, December 20, 1997
16:30	4:30 PM today

9.1.2 Delta Time

A delta time is an offset from the current date and time to a future time. The format of a delta time is:

“+[dddd-][hh:mm:ss]”

The fields are as follows:

dddd	Number of days (range from 0 to 9999)
hh	Hours (range from 0 to 23)
mm	Minutes (range from 0 to 59)
ss	Seconds (range from 0 to 59)

The following rules apply to a delta time:

- You must specify a plus (+) sign.
- You can truncate a delta time on the right.
- If you specify the number of days, include a hyphen
- You can omit fields within the time as long as you include the punctuation that separates the fields.
- If you omit the time field, the default is zero.

Examples:

“+5-“	5 days from now
“+3”	3 hours from now
“+:30”	30 minutes from now

9.1.3 Combination Time

To combine absolute and delta times, specify an absolute time followed by a delta time.

"[absolute time][+delta time]"

"[absolute time][-delta time]"

The following rules apply:

- Precede the delta time value by a plus sign for future time or a minus sign for past time.
- Enclose the entire specification in quotation marks if a plus or minus sign precedes the delta time value.
- Omit the absolute time value if you want to offset the delta time from the current date and time.
- Remember that the rules for absolute and delta time values still apply.

You can also specify **TODAY, TOMORROW** or **YESTERDAY** (the time is always midnight (00:00:00.000)) as well as **MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY** or **SUNDAY** (which calculate a forward date to the next day/date, the time is always midnight).

Examples:

"+5"	5 hours from now.
"-1"	Current time minus 1 hour. (Note that the 1 is interpreted as an hour and not a day because it is not followed by a hyphen.)
"-3-"	Current time minus 3 days. (Note the hyphen.)
"+:5"	5 minutes from now.
"-:5"	Current time minus 5 minutes.

9.2 Commands

9.2.1 CLOSE/QUEUE

The CLOSE/QUEUE command prevents further job submission.

Format:

CLOSE/QUEUE queue-name

Parameter:

queue-name

This required parameter indicates the specific queue that is to be closed.

Qualifiers:

/JS_PASSWORD[=password]
This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD
This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username
This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG
This qualifier is used to display a message when the operation is successful.

/ON=scheduler-machine-name
This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

Example:

CLOSE/QUEUE TESTQUEUE
This example closes a queue named TESTQUEUE and prevents further job submission to the queue.

9.2.2 DELETE/JOB

The DELETE/JOB command aborts or deletes a specific batch job, or in the case of a job template, the template and its instances can be aborted or deleted.

Format:

DELETE/JOB=job-entry-id/qualifiers

Qualifiers:

/INSTANCE
This qualifier will abort the executing copy of a scheduled job. A template job will remain eligible for re-execution at a later time. This qualifier is ignored if /PERMANENT is specified.
/JOB=job-entry-id
This required qualifier indicates the specific job entry via its unique entry identification that is to be aborted/deleted.
/JS_PASSWORD[=password]
This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.
/[NO]JS_SAVEPASSWORD
This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.
/JS_USERNAME=username
This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.
/LOG
This qualifier is used to display a message when the operation is successful.
/ON=scheduler-machine-name
This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).
/PERMANENT
This qualifier, if specified, indicates that the job is to be deleted. By default, job(s) are aborted and a job's history can be viewed in the Runbook.

Example:

$ DELETE/JOB=5/ON=NTSERVER1

This example aborts a job with the job id of 5 on the Job Scheduler machine NTSERVER1.

9.2.3 DELETE/QUEUE

The DELETE/QUEUE command deletes a named queue.

Format:

DELETE/QUEUE/qualifiers queue-name

Parameter:

queue-name
The name of the queue which is to be deleted. ActiveBatch requires that no jobs be present on the queue (otherwise you should use the /FLUSH qualifier).

Qualifiers:

/FLUSH
This qualifier indicates that any jobs which are held or pending are to be deleted prior to deleting the queue itself. By default, pending jobs are not deleted. A queue cannot be deleted with pending jobs.

/JS_PASSWORD[=password]
This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD
This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username
This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG
This qualifier is used to display a message when the operation is successful.

/ON=scheduler-machine-name
This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

Example:

$ DELETE/QUEUE TESTQUEUE /FLUSH

This example deletes a queue named TESTQUEUE and the /FLUSH Qualifier deletes any jobs which are held or pending.

9.2.4 FLUSH/QUEUE

This command will delete all jobs on a specified queue. Jobs can also be deleted by job state and/or be specifying the user's username.

Format:

FLUSH/QUEUE/qualifiers queue-name

Parameter:

queue-name

The name of the queue on which the flush operation is to be performed.

Qualifiers:

/BY_JOB_STATUS=(ABORTING | EXECUTING | PENDING | TIMED_RELEASE | HOLDING | COMPLETED)

This qualifier indicates which jobs by "job state" are to be deleted. By default, all jobs are deleted.

/JS_PASSWORD[=password]

This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD

This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username

This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG

This qualifier is used to display a message when the operation is successful.

/ON=scheduler-machine-name

This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

/USERNAME=username

This qualifier allows the search of jobs to be narrowed to a specific username. By default, the search is not narrowed to a specific user.

Example:

```
$ FLUSH/QUEUE TESTQUEUE
```

This example will delete all jobs that are in the queue named TESTQUEUE.

9.2.5 HOLD

This command will hold a pending job. In other words, the state of the job is changed to "Hold".

Format:

HOLD/JOB=job-entry-id/qualifiers

Qualifiers:

/JOB=job-entry-id

This required qualifier indicates the specific job entry via its unique entry identification is to be set to "Hold" status. Please note that an active executing job cannot have its state changed via this command.

/JS_PASSWORD[=password]

This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD

This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username

This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG

This qualifier is used to display a message when the operation is successful.

/ON=scheduler-machine-name

This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

Example:

```
$ HOLD/JOB=5/ON=NTSERVER1
```

This example will hold a pending or completed job with an entry id of 5 on Job Scheduler machine NTSERVER1.

9.2.6 INITIALIZE Queue

This command is used to create or initialize a new queue within the ActiveBatch system.

Format:

INITIALIZE/QUEUE/qualifiers queue-name

Parameter:

queue-name

This required parameter represents a unique name for a batch queue.

Qualifiers:

/BASE_PRIORITY=(IDLE | BELOWNORMAL | NORMAL | ABOVENORMAL | HIGH |REALTIME | number)

This qualifier represents the operating system scheduling priority assigned to the batch process at job startup. For non-Windows 2000/NT machines, a priority number would be used.

/CLOSE

This qualifier indicates whether the initial state of the queue should be to allow job submissions. If /CLOSE is specified, job submissions will not be allowed. By default, the initial queue state is OPEN. /CLOSE and /OPEN are mutually exclusive.

/CLOSE_TIME=(hh:mm-hh:mm)

This qualifier indicates the time of day that the queue is to remain closed. During the remaining portion the queue is open. The time specifier *hh:mm* represents a 24-hour time. The time range indicates the starting and ending closing time.

/DENY=(account:access…,…)

This qualifier is used to deny an account access that the account may also obtain through a group. *account* refers to the WinNT account name (or group) and *access* refers to one or more of the following keywords: EXECUTIVE, MANAGE and USER. To specify more than one level of access to be denied, separate each access with a colon. Multiple accounts may be specified by delimiting each account/access string with a comma.

/DESCRIPTION="string"

This qualifier allows a short description to be assigned and stored with the created queue.

/[NO]EVENT_LOG=keyword | \\machine

This qualifier indicates whether the Windows 2000/NT Event Log should be used for start and stop batch job messages and, if so, which machine's event log should be used. Available keywords are: Client, JobScheduler and ExecutionMachine. You may also specify a specific machine by prefacing the machine name with a \\ (2 backslashes) notation. To avoid using the Event Log specify /NOEVENT_LOG.

/EXEC_ON=machine-name

For an execution queue, this qualifier indicates the machine that submitted jobs are to be executed on.

/GENERIC=(ExecQueue1, ExecQueue2,…. ExecQueueN)

This qualifier indicates that the queue to be created is a generic queue. A generic queue points to one or more execution queues. The values specified with this qualifier are the actual execution queue names. If this qualifier is omitted, the queue is assumed to be an execution type queue.

/GRANT=(account:access,…)

This qualifier is used to associate security permissions to the queue that is to be created. By default, the owner and Administrator are given full access to the queue and the all other users are given Submit level access (this default can be changed with the SET BATCH command). To associate different permissions for this queue, you may specify one or more accounts and the level of access each account (or group) may have with this queue. *Access* is specified using one or more of the following keywords:

FULL	Complete access
EXECUTIVE	Give/Take Ownership
MANAGE	Ability to issue queue management commands
USER	Ability to SUBMIT jobs

Each keyword specified must be separated by a colon. Multiple accounts are separated by a comma.

/HOLD_ON_ERROR

This qualifier is used to establish a default for jobs which are submitted on this queue. When enabled, the job will be automatically placed in a Hold state if the job completes on error.

/JOB_LIMIT=number

This qualifier, valid for execution queues, indicates the number of simultaneous jobs that may be executed for this execution queue. The valid range is 1 through 10. By default, one is assumed.

/JS_PASSWORD[=password]

This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD

This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username

This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG

This qualifier is used to display a message when the operation is successful.

/ON=scheduler-machine-name

This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

/<u>OPEN</u>

This qualifier indicates that the queue is available for job submissions. By default, the /OPEN qualifier is assumed.

/PRIORITY=number

This qualifier indicates the default queue priority to be used when submitting a job to a queue and a priority is not specified as part of job submission. This qualifier should not be mistaken as the scheduling priority of the job (see /BASE_PRIORITY). The valid range is 1 through 99. The default is 10.

/REMOVE=(account,…)

This qualifier removes the specified accounts permissions. More than one account may be specified separated by a comma.

/SCHEDULING = (SEQUENTIAL | ROUND-ROBIN | BALANCED)

This qualifier indicates the scheduling algorithm which will be used for a Generic queue. Sequential means that jobs are submitted to the first Execution Queue listed until that queue is executing its maximum job limit. Round-Robin means that each queue, in turn, is given a job to execute until its job limits are met. Balanced indicates that a workload balancing approach be used which determines the best machine to run on based on CPU, memory and other parameters.

/[NO]START

This qualifier indicates the initial queue state after becoming initialized. /START (default) indicates that any pending jobs which are eligible to become executable should be started. /NOSTART indicates that the jobs should not be started for execution. Please note that OPEN and CLOSE qualifiers deal with the ability to submit jobs to a queue while START and NOSTART deal with the ability to execute those eligible jobs.

/STOP_TIME=(hh:mm-hh:mm)

This qualifier indicates the time of day that the queue is to remain stopped. During the remaining portion the queue is started. The time *hh:mm* represents a 24-hour time. The time range indicates the starting and ending stopped time.

/USER_CHAR=(char:datatype:value,…)

This qualifier is used to associate user characteristics with an Execution Queue. *Char* indicates a user characteristic name. This name is free form but cannot contain spaces. *Datatype* is either String or Numeric. *Value* is a string or numeric value based on the characteristic datatype. You may specify multiple characteristics by separating each with a comma.

/WORKING_FOLDER=device-directory-spec

This qualifier indicates the default working directory for jobs submitted to this queue in which a directory has not been specified.

Example:

$ INITALIZE/QUEUE TESTQUEUE /DESCRIPTION="A Test Queue"

This example creates a queue called TESTQUEUE with a description of "A Test Queue".

9.2.7 MODIFY/JOB

This command is used to modify a job's characteristics on queue.

Format:

MODIFY/JOB=job-entry-id/qualifiers

Qualifiers:

/[NO]ACTIVEBATCH_EVENT
This qualifier indicates whether the job can accept an ActiveBatch TRIGGER command. The default is that a job can accept the command.

/AFTER=combination-time
This qualifier is used to defer a job so that the job will be executed at some future time. The time specified is any valid combination time (including keywords TODAY and TOMORROW). If /AFTER is specified without a combination time specified the keyword TODAY is assumed. By default, job execution is assumed to be immediate.

/BASE_PRIORITY=(LOW | BELOWNORMAL | <u>NORMAL</u> | ABOVENORMAL | HIGH | REALTIME | number)
This qualifier represents the operating system scheduling priority assigned to the batch job. For Windows 2000/NT machines the keywords above are used. For non-Windows machines, a priority number must be specified. By default, the base priority is taken from the queue's characteristics.

/CALENDAR=calendar-name
This qualifier associates a business calendar with the job. This allows a job to be scheduled only on working days.

/CATEGORY="string"
This qualifier indicates the user category or other string value that will be kept with the job.

/CONTACT=(delivery:target[:time],…)
This qualifier is used to denote contact information for those machines/users that are to be notified when a job starts or completes execution. **Delivery** represents one of two keywords which indicate how the notification is to be occur. MAIL or MESSENGER are valid delivery keywords. **Target** represents either a machine name, if the delivery is MESSENGER, or a valid e-mail address if the delivery is MAIL. **Time** indicates either a START or FINISH job time criteria that is used to indicate when the notification is to be sent. If "time" is omitted, then FINISH is assumed. You may specify more than one contact by separating each contact specification with a comma.

/DENY=(account:access,…)
This qualifier is used to deny an account access that the account may also obtain through a group. *account* refers to the WinNT account name (or group) and *access* refers to one or more of the following keywords: READ, WRITE, DELETE, EXECUTE, TAKEOWNERSHIP, TRIGGER, TRIGGERPARAM and TRIGGERQUEUE. To specify more than one level of access to be denied, separate each access with a colon. Multiple accounts may be specified by delimiting each account/access string with a comma.

/DESCRIPTION="string"
This qualifier is used to set the pending job's description specified at job submission time.
/DOCUMENTATION="string"
This qualifier is used to associate a documentation reference with the job.
/EVERY=(MINUTE=mm,… , HOUR=hh,… , WEEKDAY=ddd,… , DAY=dd,… , MONTH=mm,…)
Note: This qualifier while supported is obsolete and has been superceded by the /SCHEDULE qualifier. This qualifier indicates that a rescheduled job entry is to be created. By default, if all keywords are omitted, a job entry is created that will execute at midnight each day. Valid keywords are as follows. Specification of multiple values or multiple keywords is supported through the use of comma delimitation.

MINUTE	valid range of 0 through 59. Default is 0.
HOUR	valid range of 0 through 23. Default is 0.
WEEKDAY	acceptable values are SUN, MON, TUE, WED, THU, FRI, SAT.
DAY	valid range of 1 through 31. A day which is specified as greater than the number of days in the month denotes the last day of the month. Default is every day of a month.
MONTH	valid range of 1 through 12. Default is every month.

The EVERY and INTERVAL qualifiers are mutually exclusive. When a DAY is specified that exceeds the actual number of days in the month, ActiveBatch will assume that the last day of the month is to be selected. If multiple DAYs are specified which exceed the actual number of days in the month, the job will be scheduled once on the last day of the month.
/[NO]FAILURESTART=(job-id,…)
This qualifier indicates one or more jobs to trigger if the job completes in failure. One or more job-id's are entered in a comma delimited format. /NOFAILURESTART removes all failure job triggers.
/GRANT=(account:access,…)
This qualifier is used to associate security permissions to the job that is to be created. To associate permissions for this job, you may specify one or more accounts and the level of access each account (or group) may have. *Access* is specified using one or more of the following keywords:

READ	Read access
WRITE	Read and Write access
DELETE	Delete access
TAKEOWNERSHIP	Take Ownership
TRIGGER	Trigger access
TRIGGERQUEUE	Trigger with Queue change access
TRIGGERPARAM	Trigger with Parameters change
EXECUTIVE	Security/Take Ownership
FULL	Complete access

Each keyword specified must be separated by a colon. Multiple accounts are separated by a comma.
/GROUP=name
This qualifier indicates the group name for this job. A group name may be used to group related jobs and perform group operations.

/[NO]HOLD[=(SUBMIT, ERROR, COMPLETION)]
This qualifier is used to set the “Hold” state. The SUBMIT keyword (the default if only /HOLD is specified) indicates the job is to enter a “Hold” state immediately upon a successful submission. The COMPLETED keyword indicates the job is to enter a “Hold” state upon job completion. One or both keywords may be specified.
/IFACTIVE=(keyword | WAIT[:time])
This qualifier indicates whether more than one instance of a scheduled job can execute. Possible keywords are: SKIP, MULTIPLE and WAIT. SKIP indicates that the scheduled instance should be skipped if an instance is already executing. MULTIPLE indicates that more than one instance of the job can run. WAIT indicates that the job should wait. You may optionally indicate how long to wait before skipping the job. SKIP is the default.
/IMMEDIATE
This qualifier when used with the /SCHEDULE qualifier indicates that the job is to run immediately.
/INTERVAL=delta-time
This qualifier indicates that a rescheduled job entry should be created. A valid delta time can be specified with the exclusion of the TODAY and YESTERDAY keywords. The delta-time specification allows for a self-relative time specification in days, hours, minutes and seconds which are added to the last submission time of the job. EVERY and INTERVAL qualifiers are mutually exclusive.
/JS_PASSWORD[=password]
This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.
/[NO]JS_SAVEPASSWORD
This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.
/JS_USERNAME=username
This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.
/LOG
This qualifier is used to display a message when the operation is successful.
/[NO[MANAGE_LOGFILES
This qualifier is used to indicate whether ActiveBatch is to centrally manage the job’s log files. If enabled, the location and name of the job’s log files are specified by ActiveBatch (you can programmatically obtain the names). If disabled, you retain the ability to locate and name the log files.
/[NO]MAX_RESTARTS=n
This qualifier is used to specify the maximum number of restarts the job may undergo. /NOMAX_RESTARTS means an unlimited number and a value specified means the job will be limited to that number of restarts. Please note that this value is applicable until the job completes.
/NAME=jobname
This qualifier is used to set the name of the submitted job. By default , the job name is set to that of the filename portion of the job file specification.

/[NO]NOTIFY[=username]
This qualifier is used to indicate that the user submitting the job wants to be notified when the job completes. If a value is set with the NOTIFY qualifier that value is a username specification. By default, NONOTIFY is assumed.
/ON=scheduler-machine-name
This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).
/[NO]OUTPUT[=file-specification]
This qualifier is used to indicate whether a log file is to be created with the batch job and if so the specification of that log file. /NOOUTPUT means that a log file isn't to be created. By default, a batch log file is produced in the current directory of the created job. The default batch log file specification is the job-file-specification with a .LOG extension. When /INTERVAL or /EVERY or /SCHEDULE is specified, the following additional information is automatically appended to the output log filename: _mmdddyyyy_hhmmss. This helps keep each individual log file separate and distinct as a batch job is rescheduled. Note: This qualifier is ignored if the /MANAGE_LOGFILES qualifier is specified.
/[NO]PARAMETER=(value1,…value8)
This qualifier is used to set the input parameters associated with a specific job. Values specified may be strings or numbers. A maximum of eight (8) values may be specified. NOPARAMETER means that no input parameters are to be associated with the job (which is the default).
/PASSWORD[="string"]
This qualifier is used to associate a password in conjunction with the current username or a username specified with the /USERNAME qualifier. If specified as a "string" value be sure to enclose the password as a quoted string to prevent the case from being changed. If a password value is not specified, the command will prompt for one (echo is disabled for security purposes).
/[NO]POST_STEP=file-specification
This qualifier indicates that a post-job step is to be performed.
/[NO]PRE_STEP=file-specification
This qualifier indicates that a pre-job step is to be performed. The pre-job step is executed prior to the main job file.
/PRIORITY=number
This qualifier is used to set the queue insertion priority of a job. The valid range is 1 through 99. By default, the queue insertion priority is taken from the queue's characteristics.
/PROCESSORMASK=n
This qualifier is used, in a multi-processing system, to denote processor affinity. The value specified should be considered a bit-mask. For example, a value of 3 means bit 0 and 1 and would denote processors zero and one.
/QUEUE_SELECTION=ANY | ALL | EVENT
This qualifier, appropriate to Generic Queues only, allows you to select whether one eligible execution queue should be used, all eligible machine queues should be used or only the machine that generated the event should be used.
/[NO]READ_ONLY
This qualifier indicates that the job's properties should be set to a "read only" state. The default is /NOREAD_ONLY.

/REMOVE=(account,…)
This qualifier removes the specified accounts permissions. More than one account may be specified separated by a comma.

/[NO]RESTART=[WAIT | FAILOVER | TERMINATE]
This qualifier is used to set the restartability of an executing job. /RESTART indicates that the job is to be automatically requeued for execution. /NORESTART indicates that the job is to be removed from the queueing system in the event of a system failure.

/[NO]RETAIN_HISTORY=days
This qualifier is used to indicate the retention period for a completed job. The /NORETAIN_HISTORY means that the job's history is to be removed when the ActiveBatch Database Purge program runs. A value specified indicates the number of days to save the history information.

/[NO]RETAIN_LOGFILES=days
This qualifier is used to indicate the retention period of the job's log files. The /NORETAIN_LOGFILES means that the job's log files are to be removed when the ActiveBatch Database Purge program runs. A value specified indicates the number of days to save the log files. Please note that this value should not be less than that specified for /RETAIN_HISTORY (otherwise it will be automatically adjusted to the same value).

/[NO]RUNONCE
This qualifier is used to specify whether the job is to run once regardless of whether the job is date/time triggered or event triggered. The default is /NORUNONCE.

/[NO]SAVE_PASSWORD
This qualifier, valid on Windows 2000/NT machines only, indicates that the password specified should be saved. /NOSAVE_PASSWORD indicates that a saved password should be removed and any specified password should not be saved.

/[NO]SCHEDULE=(name,…)
If specified this qualifier indicates that the named schedules should be associated with this job. The schedules must already exist (if not, you should use the INITIALIZE/SCHEDULE command to create one).

/SUCCESSRULE=(n | n-n | ODD | EVEN | LZERO | NTMSG)
This qualifier indicates how ActiveBatch interprets the main job step's exit code. The values specified all indicate that the job completed successfully. You may specify a single value, a range of values and/or a keyword indicating success.

/[NO]SUCCESSSTART=(job-id,…)
This qualifier indicates one or more jobs to trigger if the job completes successfully. One or more job-id's are entered in a comma delimited format.

/TAKE_OWNERSHIP
This qualifier indicates that the current user is requesting ownership of the specified job. The user must have *Executive* or *Full* access permission in order to successfully take ownership.

/[NO]TERMINATE
This qualifier indicates whether children processes should be terminated when the job exits. On Windows NT V4, children processes are identified as having the same parent id (PID). This technique is not precise. On Windows 2000, a job with this qualifier enabled, is created as a job object.

OpenVMS and Unix operating systems terminate child or subprocess trees regardless of the setting of this qualifier. By default, /NOTERMINATE is assumed.

/TIME_ZONE= CLIENT | <u>JOBSCHED</u> | UTC

This qualifier indicates the time zone which should be used by the Job Scheduler to actually schedule time triggers. By default the Job Scheduler's time zone is used.

/USERNAME=username

This qualifier is used to set the username that initially submitted the job for execution. Typically when this qualifier is used, the /PASSWORD qualifier is asserted as well.

/WORKING_FOLDER=device-directory-spec

This qualifier indicates the default working directory for this job.

Example:

$ MODIFY/JOB=5/ON=NTSERVER1/HOLD/NORESTART

This example will hold a job with an entry id of 5 on queue server machine NTSERVER1 and change the state to 'Hold" and make the job non-restartable in the event of a system failure.

9.2.8 MODIFY/QUEUE

This command is used to change a queue's characteristics.

Format:

MODIFY/QUEUE/qualifiers queue-name

Parameter:

queue-name

This required parameter represents the name of the batch queue that is to be modified.

Qualifiers:

/BASE_PRIORITY=(IDLE | BELOWNORMAL | <u>NORMAL</u> | ABOVENORMAL | HIGH |REALTIME | number)

This qualifier represents the operating system scheduling priority assigned to the batch process at job startup. For non-Windows 2000/NT machines, a priority number would be used. By default, NORMAL priority is used.

/CLOSE_TIME=(hh:mm-hh:mm)

This qualifier indicates the time of day that the queue is to remain closed. During the remaining portion the queue is open. The time specifier *hh:mm* represents a 24-hour time. The time range indicates the starting and ending closing time.

/DENY=(account:access,…)

This qualifier is used to deny an account access that the account may also obtain through a group. *account* refers to the WinNT account name (or group) and *access* refers to one or more of the following keywords: EXECUTIVE, MANAGE and USER. To specify more than one level of access to be denied, separate each access with a colon. Multiple accounts may be specified by delimiting each account/access string with a comma.

/DESCRIPTION="string"

This qualifier allows a short description to be assigned and stored with the created queue.

/[NO]EVENT_LOG=keyword | \\machine

This qualifier indicates whether the Windows 2000/NT Event Log should be used for start and stop batch job messages and, if so, which machine's event log should be used. Available keywords are: Client, JobScheduler and ExecutionMachine. You may also specify a specific machine by prefacing the machine name with a \\ (2 backslashes) notation. To avoid using the Event Log specify /NOEVENT_LOG.

/EXEC_ON=machine-name

For an execution queue, this qualifier indicates the machine that submitted jobs are to be executed on.

/GENERIC=(ExecQueue1, ExecQueue2,…. ExecQueueN)

This qualifier indicates that the queue to be created is a generic queue. A generic queue points to one or more execution queues. The values specified with this qualifier are the actual execution queue names. If this qualifier is omitted, the queue is assumed to be an execution type queue.

/GRANT=(account:access,…)

This qualifier is used to associate security permissions to the queue that is to be created. By default, the owner and Administrator are given full access to the queue and the all other users are given Submit level access (this default can be changed with the SET BATCH command). To associate different permissions for this queue, you may specify one or more accounts and the level of access each account (or group) may have with this queue. *Access* is specified using one or more of the following keywords:

FULL	Complete access
EXECUTIVE	Give/Take Ownership
MANAGE	Ability to issue queue management commands
USER	Ability to SUBMIT jobs

Each keyword specified must be separated by a colon. Multiple accounts are separated by a comma.

/JOB_LIMIT=number

This qualifier, valid for execution queues, indicates the number of simultaneous jobs that may be executed for this execution queue. The valid range is 1 through 10. By default, one is assumed.

/JS_PASSWORD[=password]

This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD

This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username

This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG

This qualifier is used to display a message when the operation is successful.

/ON=scheduler-machine-name

This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

/PRIORITY=number

This qualifier indicates the default queue priority to be used when submitting a job to a queue and a priority is not specified as part of job submission. This qualifier should not be mistaken as the scheduling priority of the job (see /BASE_PRIORITY). The valid range is 1 through 99. The default is 10.

/REMOVE=(account,…)
This qualifier removes the specified accounts permissions. More than one account may be specified separated by a comma.

/SCHEDULING = (SEQUENTIAL | ROUND-ROBIN | BALANCED)
This qualifier indicates the scheduling algorithm which will be used for a Generic queue. Sequential means that jobs are submitted to the first Execution Queue listed until that queue is executing its maximum job limit. Round-Robin means that each queue, in turn, is given a job to execute until its job limits are met. Balanced indicates that a workload balancing approach be used which determines the best machine to run on based on CPU, memory and other parameters.

/STOP_TIME=(hh:mm-hh:mm)
This qualifier indicates the time of day that the queue is to remain stopped. During the remaining portion the queue is started. The time *hh:mm* represents a 24-hour time. The time range indicates the starting and ending stopped time.

/TAKE_OWNERSHIP
This qualifier indicates that the current user is requesting ownership of the specified queue. The user must have *Executive* or *Full* access permission in order to successfully take ownership.

/USER_CHAR=(char:datatype:value,…)
This qualifier is used to associate user characteristics with an Execution Queue. *Char* indicates a user characteristic name. This name is free form but cannot contain spaces. *Datatype* is either string or numeric. *Value* is a string or numeric value based on the characteristic datatype. You may specify multiple characteristics by separating each with a comma.

/WORKING_FOLDER=device-directory-spec
This qualifier indicates the default working directory for jobs submitted to this queue in which a directory has not been specified.

Example:

```
$ MODIFY/QUEUE TESTQUEUE /BASE_PRIORITY=HIGH -
        /GRANT=EVERYONE:MANAGE -
        /JOB_LIMIT=5
```

This example will modify a queue named TESTQUEUE and set the O/S priority to HIGH, grant the *Everyone* Group Manage permissions, set the job limit to 5.

9.2.9 OPEN/QUEUE

The OPEN/QUEUE command allows job submission.

Format:

OPEN/QUEUE queue-name

Parameter:

queue-name

This required parameter indicates the specific queue that is to be opened.

Qualifiers:

/JS_PASSWORD[=password]

This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD

This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username

This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG

This qualifier is used to display a message when the operation is successful.

/ON=scheduler-machine-name

This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

Example:

$ OPEN/QUEUE TESTQUEUE

This example opens the queue named TESTQUEUE and allows new job submission.

9.2.10 PAUSE

This command will pause an executing job. Note: On Windows 2000/NT machines, only the primary thread is suspended. For a multi-threaded process this may not necessarily produce the expected results.

Format:

PAUSE/JOB=job-entry-id/qualifiers

Qualifiers:

/JOB=job-entry-id
This required qualifier indicates the specific job entry via its unique entry identification that is to be "Paused". Please note that this command is only applicable to an executing job.

/JS_PASSWORD[=password]
This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD
This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username
This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG
This qualifier is used to display a message when the operation is successful.

/ON=scheduler-machine-name
This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

Example:

$ PAUSE/JOB=5/ON=NTSERVER1

This example will pause an executing job with an entry id of 5 on queue server NTSERVER1.

9.2.11 RELEASE

This command will release a held job.

Format:

RELEASE/JOB=job-entry=id/qualifiers

Qualifiers:

/JOB=job-entry-id
This required qualifier indicates the specific job entry via its unique entry identification that is currently held and will now be released for possible execution.

/JS_PASSWORD[=password]
This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD
This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username
This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG
This qualifier is used to display a message when the operation is successful.

/ON=scheduler-machine-name
This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

Example:

$ RELEASE/JOB=5/ON=NTSERVER1

This example will release a held job with an entry id of 5 on queue server machine NTSERVER1.

9.2.12 REQUEUE

This command will requeue (or move) a job to another queue.

Format:

RELEASE/JOB=job-entry=id/qualifiers

Qualifiers:

/JOB=job-entry-id
This required qualifier indicates the specific job entry via its unique entry identification that is to be moved to another queue.

/JS_PASSWORD[=password]
This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD
This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username
This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG
This qualifier is used to display a message when the operation is successful.

/ON=scheduler-machine-name
This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

/QUEUE=new-queue-name
This qualifier indicates the destination queue that this job is to be moved to.

Example:

$ REQUEUE/JOB=15/QUEUE=MASTER

This example will requeue job 15 to queue MASTER.

9.2.13 RESTART

This command will abort and then restart an already active batch job.

Format:

RESTART/qualifiers

Qualifiers:

/JOB=job-entry-id

This required qualifier indicates the specific job entry via its unique entry identification that is currently active and will now be restarted.

/JS_PASSWORD[=password]

This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD

This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username

This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG

This qualifier is used to display a message when the operation is successful.

/ON=scheduler-machine-name

This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

Example:

$ RESTART/JOB=5/ON=NTSERVER1

This command will abort and then restart the job with an entry id of 5 on a machine named NTSERVER1.

9.2.14 RESUME

This command will resume a paused job.

Format:

RESUME/JOB=job-entry-id/qualifiers

Qualifiers:

/JOB=job-entry-id
This required qualifier indicates the specific job entry via its unique entry identification that a paused job is to be resumed for continued execution.

/JS_PASSWORD[=password]
This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD
This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username
This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG
This qualifier is used to display a message when the operation is successful.

/ON=scheduler-machine-name
This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

Example:

$ RESUME/JOB=5/ON=NTSERVER1

This example will resume a paused job with an entry id of 5 on Job Scheduler machine NTSERVER1.

9.2.15 SET BATCH

This command allows an Administrator to establish defaults for an ActiveBatch user as well as change ActiveBatch Administrator status.

Format:

SET BATCH/qualifiers

Qualifiers:

/DENY=(account:access…,…)

This qualifier is used to deny an account access that the account may also obtain through a group. *account* refers to the WinNT account name (or group) and *access* refers to one or more of the following keywords: EXECUTIVE, MANAGE and USER. To specify more than one level of access to be denied, separate each access with a colon. Multiple accounts may be specified by delimiting each account/access string with a comma.

/GRANT=(account:access,…)

This qualifier is used to associate security permissions to the queue that is to be created. By default, the owner and Administrator are given full access to the queue and the all other users are given Submit level access (this default can be changed with the SET BATCH command). To associate different permissions for this queue, you may specify one or more accounts and the level of access each account (or group) may have with this queue. *Access* is specified using one or more of the following keywords:

FULL	Complete access
EXECUTIVE	Give/Take Ownership
MANAGE	Ability to issue queue management commands
USER	Ability to SUBMIT jobs

Each keyword specified must be separated by a colon. Multiple accounts are separated by a comma.

/JS_PASSWORD[=password]

This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD

This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username

This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/ON=scheduler-machine-name

This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

/[NO]QUEUE=queue-name
This qualifier allows you to designate a default Queue to be used when a SUBMIT command is issued. By default, the /QUEUE qualifier is required to be specified with the SUBMIT command. Negating the qualifier causes the default to be removed.
/REMOVE=(account,…)
This qualifier removes the specified accounts permissions. More than one account may be specified separated by a comma.
/[NO]SCHEDULER=machine
This qualifier allows you to designate a default Job Scheduler machine to be used when invoking other command-line utilities. By default the local machine is assumed. Negating the qualifier causes the default to be removed.

9.2.16 SHOW JOB

This command will display batch jobs based on various search parameters.

Format:

SHOW JOB/qualifiers

Qualifiers:

/ALL_JOBS
By default, only jobs submitted by the owner are eligible for display. This qualifier indicates that all batch jobs, regardless of who submitted the job, are eligible for possible display.

/BRIEF
This qualifier indicates that only very terse information be displayed for each job. BRIEF is the default.

/BY_JOB_STATUS=(DEFERRED | EXECUTING | HELD | PENDING)
This qualifier allows you to further refine the search to those jobs in the above specified job states.

/EXECUTION_QUEUE
This qualifier allows the search for eligible jobs to be further refined to those on execution queues.

/FULL
This qualifier is used to display complete information on all jobs which are to be displayed. /BRIEF is the default.

/GENERIC
This qualifier displays jobs that are on a generic queue. EXECUTION_MACHINE and GENERIC qualifiers are mutually exclusive.

/JOB=job-entry-id
This required qualifier indicates the specific job entry via its unique entry identification is to be set to "Hold" status. Please note that an active executing job cannot have its state changed via this command.

/JS_PASSWORD[=password]
This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD
This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username
This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/ON=scheduler-machine-name
This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

/OUTPUT=FileSpecification
This qualifier allows the command display to be redirected to a disk file. By default, the output of this command is sent to Standard Out (normally the console window).

/USERNAME=username
This qualifier allows the search for eligible jobs to be further refined to a specific username. By default, the username of this command is used for job selection. ALL_JOBS and USERNAME are mutually exclusive.

Example:

$ SHOW JOB /FULL /JOB=6

```
Job 6 submitted to Queue TESTQUEUE on TEST by XLNTUSER
Job Name : TEST2        Status : Retained : CFFD00B5
Priority : 10             File : C:\TESTXCP\TEST2.XCP
Command Shell :   XLNT Log File : C:\TESTXCP\TEST2.LOG

Characteristics : Job Restart disabled, Notification enabled for
TEST:XLNTUSER via NT Messenger Service, Log File printing disabled, Log
File deleted
```

This example shows a batch job with an ID of 6 on a machine called TEST and shows all information about that job.

9.2.17 SHOW QUEUE

This command will display batch jobs based on various search parameters.

Format:

SHOW QUEUE/qualifiers [queue-name]

Parameter:

queue-name

This optional parameter represents a specific queue name to display. If omitted, all queues are eligible for display.

Qualifiers:

/ALL_JOBS

By default, only jobs submitted by the owner are eligible for display. This qualifier indicates that all batch jobs, regardless of who submitted the job, are eligible for possible display.

/<u>BRIEF</u>

This qualifier indicates that only very terse information be displayed for each job. BRIEF is the default.

/EXECUTION_MACHINE=machine-name

This qualifier allows the search for eligible jobs to be further refined to those execution queues which are associated with a given machine.

/FULL

This qualifier is used to display complete information on all jobs which are to be displayed. /BRIEF is the default.

/GENERIC

This qualifier displays jobs that are on a generic queue. EXECUTION_MACHINE and GENERIC qualifiers are mutually exclusive.

/JS_PASSWORD[=password]

This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD

This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username

This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/ON=scheduler-machine-name

This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

/OUTPUT=FileSpecification
This qualifier allows the command display to be redirected to a disk file. By default, the output of this command is sent to Standard Out (normally the console window).
/USERNAME=username
This qualifier allows the search for eligible jobs to be further refined to a specific owner. By default, the username of this command is used for job selection. ALL_JOBS and OWNER are mutually exclusive.

Example:

$ SHOW QUEUE TESTQUEUE /FULL
```
Execution Queue TESTQUEUE on TEST, owned by XLNTUSER
Status : Started, Opened  Job Retention : Failed Jobs
Base Priority  : NORMAL  Queue Priority : 10
Command Shell : XLNT          Job Limit :  1
Description:
        A Test Queue

Account           Permissions
-------           -----------
Administrators    Granted: EXECUTIVE, MANAGE, USER
SYSTEM            Granted: EXECUTIVE, MANAGE, USER
Everyone          Granted: USER
DOMAIN\XLNTUSER   Granted: EXECUTIVE, MANAGE, USER

Jobs on Queue TESTQUEUE
        No Jobs on Queue
```

This example shows the queue called TESTQUEUE with complete information on all jobs.

9.2.18 START/QUEUE

The START/QUEUE command allows a queue to be started for job execution.

Format:

START/QUEUE queue-name

Parameter:

queue-name

This required parameter indicates the specific queue that is to be started.

Qualifiers:

/JS_PASSWORD[=password]

This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD

This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username

This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG

This qualifier is used to display a message when the operation is successful.

/ON=scheduler-machine-name

This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

Example:

$ START/QUEUE TESTQUEUE

This example starts the queue named TESTQUEUE for job execution.

9.2.19 STOP/QUEUE

The STOP/QUEUE command prevents further job execution on a queue basis and/or aborts a currently executing job.

Format:

STOP/QUEUE queue-name

Parameter:

queue-name

This required parameter indicates the specific queue that is to be stopped.

Qualifiers:

/<u>AFTER_EXECUTION</u>

This qualifier indicates that the currently executing job should be allowed to complete prior to the queue being stopped. AFTER_EXECUTION is the default.

/CLOSE

This qualifier indicates that the queue is to be closed for job submission.

/IMMEDIATE

This qualifier indicates that the currently executing job is to be aborted immediately. By default, the job is allowed to complete (see AFTER_EXECUTION).

/JS_PASSWORD[=password]

This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD

This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username

This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG

This qualifier is used to display a message when the operation is successful.

/ON=scheduler-machine-name

This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

Example:

$ STOP/QUEUE TESTQUEUE /IMMEDIATE

This example stops the queue named TESTQUEUE immediately rather than waiting for the jobs to execute.

9.2.20 SUBMIT

The SUBMIT command is used to submit a job for batch execution.

Format:

SUBMIT/qualifiers job-file-specification

Parameter:

job-file-specification

This required parameter is a file specification which will represent the job. The file specification may be an executable (.EXE) or any file which contains a known file association.

Qualifiers:

/[NO]ACTIVEBATCH_EVENT

This qualifier indicates whether the job can accept an ActiveBatch TRIGGER command. The default is that a job can accept the command.

/AFTER=combination-time

This qualifier is used to defer a job so that the job will be executed at some future time. The time specified is any valid combination time (including keywords TODAY and TOMORROW). If /AFTER is specified without a combination time specified the keyword TODAY is assumed. By default, job execution is assumed to be immediate.

/BASE_PRIORITY=(LOW | BELOWNORMAL | <u>NORMAL</u> | ABOVENORMAL | HIGH | REALTIME | number)

This qualifier represents the operating system scheduling priority assigned to the batch job. For Windows 2000/NT machines the keywords above are used. For non-Windows machines, a priority number must be specified. By default, the base priority is taken from the queue's characteristics.

/CALENDAR=calendar-name

This qualifier associates a business calendar with the job. This allows a job to be scheduled only on working days.

/CATEGORY="string"

This qualifier indicates the user category or other string value that will be kept with the job.

/CONTACT=(delivery:target[:time],…)

This qualifier is used to denote contact information for those machines/users that are to be notified when a job starts or completes execution. **Delivery** represents one of two keywords which indicate how the notification is to be occur. MAIL or MESSENGER are valid delivery keywords. **Target** represents either a machine name, if the delivery is MESSENGER, or a valid e-mail address if the delivery is MAIL. **Time** indicates either a START or FINISH job time criteria that is used to indicate when the notification is to be sent. If "time" is omitted, then FINISH is assumed. You may specify more than one contact by separating each contact specification with a comma.

/COPY=(PRE, POST, MAIN)

This qualifier is used to indicate which batch job file steps should be copied to the selected Execution Machine. The keywords PRE, POST and MAIN indicate which batch file steps are to be copied. By default, no batch files are copied.

/DENY=(account:access,…)
This qualifier is used to deny an account access that the account may also obtain through a group. *account* refers to the WinNT account name (or group) and *access* refers to one or more of the following keywords: READ, WRITE, DELETE, EXECUTE, TAKEOWNERSHIP, TRIGGER, TRIGGERPARAM and TRIGGERQUEUE. To specify more than one level of access to be denied, separate each access with a colon. Multiple accounts may be specified by delimiting each account/access string with a comma.
/DESCRIPTION=”string”
This qualifier is used to set the pending job’s description specified at job submission time.
/DOCUMENTATION=”string”
This qualifier is used to associate a documentation reference with the job.
/EVERY=(MINUTE=mm,… , HOUR=hh,… , WEEKDAY=ddd,… , DAY=dd,… , MONTH=mm,…)
Note: This qualifier while supported is obsolete and has been superceded by the /SCHEDULE qualifier. This qualifier indicates that a rescheduled job entry is to be created. By default, if all keywords are omitted, a job entry is created that will execute at midnight each day. Valid keywords are as follows. Specification of multiple values or multiple keywords is supported through the use of comma delimitation.

MINUTE	valid range of 0 through 59. Default is 0.
HOUR	valid range of 0 through 23. Default is 0.
WEEKDAY	acceptable values are SUN, MON, TUE, WED, THU, FRI, SAT.
DAY	valid range of 1 through 31. A day which is specified as greater than the number of days in the month denotes the last day of the month. Default is every day of a month.
MONTH	valid range of 1 through 12. Default is every month.

The EVERY and INTERVAL qualifiers are mutually exclusive. When a DAY is specified that exceeds the actual number of days in the month, ActiveBatch will assume that the last day of the month is to be selected. If multiple DAYs are specified which exceed the actual number of days in the month, the job will be scheduled once on the last day of the month.
/FAILURESTART=(job-id,…)
This qualifier indicates one or more jobs to trigger if the job completes in failure. One or more job-id’s are entered in a comma delimited format.
/GRANT=(account:access,…)
This qualifier is used to associate security permissions to the job that is to be created. To associate permissions for this job, you may specify one or more accounts and the level of access each account (or group) may have. *Access* is specified using one or more of the following keywords:

READ	Read access
WRITE	Read and Write access
DELETE	Delete access
TAKEOWNERSHIP	Take Ownership
TRIGGER	Trigger access
TRIGGERQUEUE	Trigger with Queue change access
TRIGGERPARAM	Trigger with Parameters change

EXECUTIVE	Security/Take Ownership
FULL	Complete access

Each keyword specified must be separated by a colon. Multiple accounts are separated by a comma.

/GROUP=name

This qualifier indicates the group name for this job. A group name may be used to group related jobs and perform group operations.

/HOLD[=(<u>SUBMIT</u>, ERROR, COMPLETION)]

This qualifier is used to set the initial "Hold" state of a job. If this qualifier is not specified, the job will become eligible for execution based on its position in an execution queue. The SUBMIT keyword (the default if only /HOLD is specified) indicates the job is to enter a "Hold" state immediately upon a successful submission. The ERROR keyword indicates that the job is to enter a "Hold" state if the job completes in error. The COMPLETION keyword indicates the job is to enter a "Hold" state upon job completion (regardless of whether the job was successful or not). One or more keywords may be specified.

/IFACTIVE=(keyword | WAIT[:time])

This qualifier indicates whether more than one instance of a scheduled job can execute. Possible keywords are: SKIP, MULTIPLE and WAIT. SKIP indicates that the scheduled instance should be skipped if an instance is already executing. MULTIPLE indicates that more than one instance of the job can run. WAIT indicates that the job should wait. You may optionally indicate how long to wait before skipping the job. SKIP is the default.

/IMMEDIATE

This qualifier when used with the /SCHEDULE qualifier indicates that the job is to run immediately.

/INTERACTIVE=(<u>NORMAL</u> | MAXIMIZED | MINIMIZED)

This qualifier indicates that the job is to be executed interactively on the desktop. The keywords NORMAL, MAXIMIZED and MINIMIZED indicate the starting state of the visible window. If the qualifier is specified and the keyword is omitted, NORMAL is assumed.

/INTERVAL=delta-time

This qualifier indicates that a rescheduled job entry should be created. A valid delta time can be specified with the exclusion of the TODAY and YESTERDAY keywords. The delta-time specification allows for a self-relative time specification in days, hours, minutes and seconds which are added to the last submission time of the job. EVERY and INTERVAL qualifiers are mutually exclusive.

/JS_PASSWORD[=password]

This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD

This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username

This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG
This qualifier is used to display a message when the operation is successful.
/[NO[MANAGE_LOGFILES
This qualifier is used to indicate whether ActiveBatch is to centrally manage the job's log files. If enabled, the location and name of the job's log files are specified by ActiveBatch (you can programmatically obtain the names). If disabled, you retain the ability to locate and name the log files.
/[NO]MAX_RESTARTS=n
This qualifier is used to specify the maximum number of restarts the job may undergo. /NOMAX_RESTARTS means an unlimited number and a value specified means the job will be limited to that number of restarts. Please note that this value is applicable until the job completes.
/NAME=jobname
This qualifier is used to set the name of the submitted job. By default , the job name is set to that of the filename portion of the job file specification.
/[NO]NOTIFY[=username]
This qualifier is used to indicate that the user submitting the job wants to be notified when the job completes. If a value is set with the NOTIFY qualifier that value is a username specification. By default, NONOTIFY is assumed. Note: This qualifier has been made obsolete by the /CONTACT qualifier. The equivalent syntax is /CONTACT=machine-name:MESSENGER:FINISH.
/ON=scheduler-machine-name
This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).
/[NO]OUTPUT[=file-specification]
This qualifier is used to indicate whether a log file is to be created with the batch job and if so the specification of that log file. /NOOUTPUT means that a log file isn't to be created. By default, a batch log file is produced in the current directory of the created job. The default batch log file specification is the job-file-specification with a .LOG extension. When /INTERVAL or /EVERY or /SCHEDULE is specified, the following additional information is automatically appended to the output log filename: _mmdddyyyy_hhmmss. This helps keep each individual log file separate and distinct as a batch job is rescheduled. Note: This qualifier is ignored if the /MANAGE_LOGFILES qualifier is specified.
/[NO]PARAMETER=(value1,…value8)
This qualifier is used to set the input parameters associated with a specific job. Values specified may be strings or numbers. A maximum of eight (8) values may be specified. NOPARAMETER means that no input parameters are to be associated with the job (which is the default).
/PASSWORD[="string"]
This qualifier is used to associate a password in conjunction with the current username or a username specified with the /USERNAME qualifier. If specified as a "string" value be sure to enclose the password as a quoted string to prevent the case from being changed. If a password value is not specified, the command will prompt for one (echo is disabled for security purposes).
/POST_STEP=file-specification
This qualifier indicates that a post-job step is to be performed.
/PRE_STEP=file-specification
This qualifier indicates that a pre-job step is to be performed. The pre-job step is executed prior to the main job file.

/PRIORITY=number
This qualifier is used to set the queue insertion priority of a job. The valid range is 1 through 99. By default, the queue insertion priority is taken from the queue's characteristics.

/PROCESSORMASK=n
This qualifier is used, in a multi-processing system, to denote processor affinity. The value specified should be considered a bit-mask. For example, a value of 3 means bit 0 and 1 and would denote processors zero and one.

/QUEUE=queue-name
This qualifier is used to indicate the queue that the job is to be submitted on. This qualifier is required unless a default queue has been established using the SET BATCH command.

/QUEUE_SELECTION=ANY | ALL | EVENT
This qualifier, appropriate to Generic Queues only, allows you to select whether one eligible execution queue should be used, all eligible machine queues should be used or only the machine that generated the event should be used.

/[<u>NO</u>]READ_ONLY
This qualifier indicates that the job's properties should be set to a "read only" state. The default is /NOREAD_ONLY.

/REMOVE=(account,…)
This qualifier removes the specified accounts permissions. More than one account may be specified separated by a comma.

/[NO]RESTART=[WAIT | FAILOVER | TERMINATE]
This qualifier is used to set the restartability of an executing job. /RESTART indicates that the job is to be automatically requeued for execution. /NORESTART indicates that the job is to be removed from the queueing system in the event of a system failure.

/[NO]RETAIN_HISTORY=days
This qualifier is used to indicate the retention period for a completed job. The /NORETAIN_HISTORY means that the job's history is to be removed when the ActiveBatch Database Purge program runs. A value specified indicates the number of days to save the history information.

/[NO]RETAIN_LOGFILES=days
This qualifier is used to indicate the retention period of the job's log files. The /NORETAIN_LOGFILES means that the job's log files are to be removed when the ActiveBatch Database Purge program runs. A value specified indicates the number of days to save the log files. Please note that this value should not be less than that specified for /RETAIN_HISTORY (otherwise it will be automatically adjusted to the same value).

/[<u>NO</u>]RUNONCE
This qualifier is used to specify whether the job is to run once regardless of whether the job is date/time triggered or event triggered. The default is /NORUNONCE.

/[NO]SAVE_PASSWORD
This qualifier, valid on Windows 2000/NT machines only, indicates that the password specified should be saved. /NOSAVE_PASSWORD indicates that a saved password should be removed and any specified password should not be saved.

/SCHEDULE=(name,…)
If specified this qualifier indicates that the named schedules should be associated with this job. The schedules must already exist (if not, you should use the INITIALIZE/SCHEDULE command to create one).

/SUCCESS_RULE=(n | n-n | ODD | EVEN | LZERO | NTMSG,…)
This qualifier indicates how ActiveBatch interprets the main job step's exit code. The values specified all indicate that the job completed successfully. You may specify a single value, a range of values and/or a keyword indicating success.

/SUCCESS_START=(job-id,…)
This qualifier indicates one or more jobs to trigger if the job completes successfully. One or more job-id's are entered in a comma delimited format.

/TAKE_OWNERSHIP
This qualifier indicates that the current user is requesting ownership of the specified job. The user must have *Executive* or *Full* access permission in order to successfully take ownership.

/[NO]TERMINATE
This qualifier indicates whether children processes should be terminated when the job exits. On Windows NT V4, children processes are identified as having the same parent id (PID). This technique is not precise. On Windows 2000, a job with this qualifier enabled, is created as a job object. OpenVMS and Unix operating systems terminate child or subprocess trees regardless of the setting of this qualifier. By default, /NOTERMINATE is assumed.

/TIME_ZONE= CLIENT | <u>JOBSCHED</u> | UTC
This qualifier indicates the time zone which should be used by the Job Scheduler to actually schedule time triggers. By default the Job Scheduler's time zone is used.

/USERNAME=username
This qualifier is used to set the username that initially submitted the job for execution. Typically when this qualifier is used, the /PASSWORD qualifier is asserted as well.

/WORKING_FOLDER=device-directory-spec
This qualifier indicates the default working directory for this job.

Example:

```
$ SUBMIT C:\TEST\TEST.XCP /QUEUE=TESTQUEUE –
        /NAME=TESTJOB /NOOUTPUT /USERNAME="XLNTUSER" /PASSWORD="12345"
```

This example will submit a batch job with a procedure name of C:\TEST\TEST.XCP, to a queue named TESTQUEUE, with a jobname of "TESTJOB" and will not send the output to a Logfile. It will also submit it under a username of XLNTUSER and a password of 12345.

9.2.21 SYNCHRONIZE

The Synchronize command allows the issuer to wait for the completion of an identified and running batch job.

Format:

SYNCHRONIZE/qualifiers

Qualifiers:

/JOB=entry-identification
This qualifier identifies a job through its unique job entry identification number assigned when the job is submitted to a queue. If you specify both /JOB and the *job-name* parameter, /JOB will be used.

/JS_PASSWORD[=password]
This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD
This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username
This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG
This qualifier is used to display a message when the operation is successful.

/ON=scheduler-machine-name
This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

Example:

```
$ SYNCHRONIZE /JOB=6
```

This example causes the issuer to wait for the completion of batch job entry id number 6.

9.2.22 TRIGGER

The Trigger command allows the issuer to trigger an existing job for immediate execution. You must have “Trigger” access permission to use this command.

Format:

TRIGGER/qualifiers

Qualifiers:

/IGNORE=(DEPENDENCIES, CALENDAR, EXLCUSION)
This qualifier indicates that the job is to ignore certain associated restrictions. The keyword restrictions are: DEPENDENCIES, CALENDAR and EXCLUSION. You may specify one or more qualifiers. By default, all restrictions are enforced. This qualifier is intended to be used when an operator or someone knowledgeable about the job and system intentionally wants to force the job to execute.

/JOB=entry-identification
This qualifier identifies a job through its unique job entry identification number assigned when the job is submitted to a queue. If you specify both /JOB and the *job-name* parameter, /JOB will be used.

/JS_PASSWORD[=password]
This qualifier is used in combination with the /JS_USERNAME and, optionally, /JS_SAVEPASSWORD to specify a password to be used with the value of /JS_USERNAME to form an authentication credential to communicate with the Job Scheduler machine.

/[NO]JS_SAVEPASSWORD
This qualifier indicates that the password specified with /JS_PASSWORD is to be stored for future authentication purposes. You can specify this qualifier only when /JS_PASSWORD is also specified. To delete a stored password specify /NOJS_SAVEPASSWORD.

/JS_USERNAME=username
This qualifier is used in combination with ./JS_PASSWORD to form an authentication credential used to communicate with the Job Scheduler machine.

/LOG
This qualifier is used to display a message when the operation is successful.

/NAME=job-name
This qualifier represents a unique job name that is to be triggered for execution. If /JOB is also specified, this qualifier is ignored.

/ON=scheduler-machine-name
This qualifier allows the command to be directed to the named Job Scheduler machine. You may specify any valid machine name or IP address (depending on the protocols your machine supports).

/PARAMETERS=(param1,…,param8)
This qualifier allows you to set input parameters when the job instance is created. You must hold “Trigger with Parameters” job access permissions to specify this qualifier.

/QUEUE=queue-name
This qualifier is used to explicitly create a job instance on the specified queue. You must hold “Trigger with Queue” job access permission to set this qualifier.

10 ACTIVEBATCH SCRIPTING AND PROGRAMMING

The ActiveBatch Client sub-system is Microsoft COM based. This means that ActiveBatch is a collection of objects, properties and methods. ActiveBatch allows you to use any object-oriented language (and scripting language) to directly manipulate these objects.

To learn more about the programmatic aspects of ActiveBatch, we invite you to read the "ActiveBatch Developer's Guide".

10.1 ActiveBatch Object Topology

Figure 132 is a drawing that visually describes the different ActiveBatch objects and their relationship.

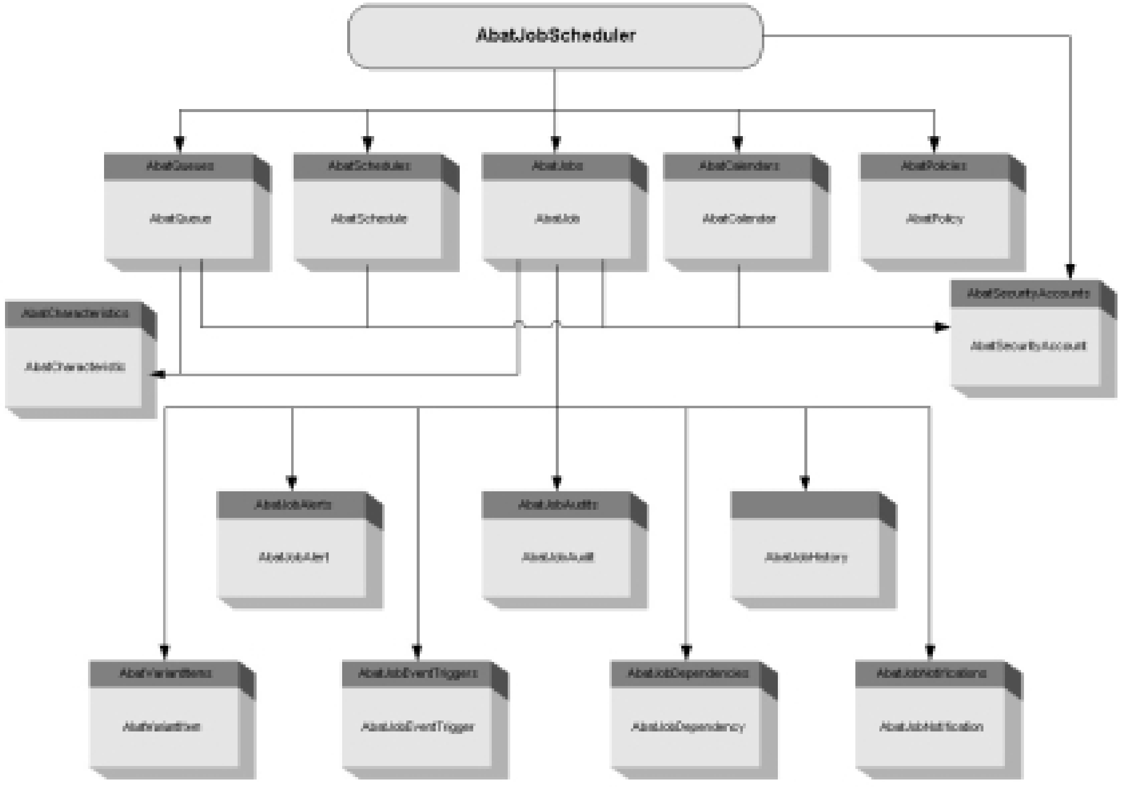


Figure 132 - ActiveBatch Object Relationship

11 ACTIVEBATCH MESSAGES

The following tables list all the pertinent ActiveBatch messages. The message codes follow the Windows 2000/NT convention and contain the severity as part of the message code. In addition, the messages themselves contain a facility code "ABAT", a severity code ("I", "W","E") and the message name followed by the text of the message. All ActiveBatch messages are contained in the ABATMSG.DLL file which can be found in the ActiveBatch installation directory.

Please note that the tables do not contain every ActiveBatch message but rather the messages which a user can typically expect to see. The first table lists all the messages in alphabetical order, the second table lists the messages using the numeric value (in hex).

Message	**Value (hex)**	**Description/Action**
ABAT-E-ABORTENTRY, Entry has been aborted	CFFD00C5	A job has been marked as 'aborted'.
ABAT-E-ALLOCFAIL, Failure to allocate memory	CFFD0008	This is an internal error usually indicating that insufficient hardware or software resources are available to satisfy a memory request.
ABAT-E-ASSOCNOTFND, Failure to determine file association	CFFD015E	The Execution Agent was unable to find the job file's extension to program association. The job was not run.
ABAT-E-AUDPURGEFAIL, Failed to purge Audit ID %1!d!	CFFD01EB	This DBPurge message can appear in the NT Application Event Log and indicates that DBPurge was unable to remove old audit messages from a specific job.
ABAT-E-BADPARAM, Bad parameter received	CFFD000B	This is a general error indicating that a bad parameter was specified. Depending on the context you should specify a legal value or property and try again.
ABAT-E-BATCHSTARTERR, unable to launch Batch Start job	CFFD01CA	A job completed and had a Batch Start (Success or Failure) action associated with it. That action could not be performed. Additional information accompanies this message indicating more specifically what the problem was.

Message	Value (hex)	Description/Action
ABAT-E-CONNECTFAIL, Failure to connect to Server, %1!s!	CFFD0005	This is a general non-specific error indicating that the client was unable to connect to a server machine. Please check the machine name or IP address and try again.
ABAT-E-COPYFILEERROR, An error occurred while copying the file '%1!s!' to the Job Scheduler because the file cannot be found or there was an error opening/reading the file.	CFFD01A7	This message can occur if the "Copy File" feature is enabled for a Main, Pre or Post file. As the message indicates an error occurred while attempting to copy the file to the Job Scheduler.
ABAT-E-CRELOGFILE, Failure to create log file	CFFD015F	The Execution Agent was unable to create the batch job's log file. Typically, the log file specification is not valid or proper OR the main job script file specification is also not valid. Please check the log/script file specification and ensure it can be created from the Execution machine.
ABAT-E-DBPURGEFAIL, An error occurred while running the DBPurge process. Please examine the NT Event Log/DBPurge Log for further details	CFFD01EA	Self-explanatory. This error message concerns the running of the DBPurge facility to prune unneeded jobs from the ActiveBatch system.
ABAT-E-DEPENDFAIL, dependency failure	CFFD01AE	The Batch Job failed because a job dependency could not be satisfied.
ABAT-E-DISCONFAIL, Failure to disconnect from server	CFFD000E	An error occurred disconnecting from a Job Scheduler machine. This is typically not something to be overly concerned about since the error occurs on the disconnect itself.
ABAT-E-DUPLICATE_KEY_VALUE, Duplicate key value	CFFD0070	This error is returned when an attempt has been made to create an object with a duplicate key (2 objects with the same name).
ABAT-E-DUPLICFND, A duplicate license was found. Please ensure that all licenses are valid and unique.	CFFD01D7	This serious error usually indicates some level of license tampering. Please run the ABATLIC program to obtain more information.
ABAT-E-EMGRCONNECTFAIL, Failure to connect to Execution Queue Manager	CFFD0007	The Job Scheduler was unable to connect to the Execution Agent machine. Please check the machine name or IP address and try again; the

Message	Value (hex)	Description/Action
		machine may not be on the network; the machine may not be running RPC or DCE services.
ABAT-E-EMGRFAIL, Network connection to the execution machine has been lost	CFFD0033	Self-explanatory. Depending on network conditions this problem may resolve itself as the machine is re-discovered.
ABAT-E-EXCCPUTIME, job aborted exceeding expected cpu time	CFFD01C5	The Execution Agent has set this job's exit status code because of an overrun for CPU time and has aborted the job.
ABAT-E-EXCRUNTIME, job aborted exceeding expected run time	CFFD01C4	The Execution Agent has set this job's exit status code because of an overrun for elapsed run time and has aborted the job.
ABAT-E-EXECABORT, job aborted due to Execution Agent failure	CFFD01D1	A job was aborted when an Execution Agent/machine failed. The job was not marked as being restartable.
ABAT-E-FAILURE, Failure in Batch Queue Management System operation	CFFD0002	A non-specific error occurred. The error is typically either internal or transient. Please try the operation again.
ABAT-E-FILESIZEERROR, The file being copied is larger than the system policy allows for	CFFD01A8	The ActiveBatch Administrator has prevented your file from being copied to the Execution Machine through the System policy CopyScriptMax
ABAT-E-GETOWNER, unable to obtain object owner	CFFD01E5	ActiveBatch is unable to retrieve the owner of the object. This error may appear if the domain of the owner's account is not available.
ABAT-E-IMPERSONERR, Failure to impersonate client	CFFD00CF	This error can be returned if you attempt to connect to a Job Scheduler while the Scheduler is still in the process of starting. Please wait a moment and try again.
ABAT-E-INITQFAIL, Failure to initialize a queue	CFFD000D	The operation to initialize a queue failed. The queue was not created. A more specific error usually accompanies this error.
ABAT-E-INITSD, Failure to initialize ABAT Security Descriptor	CFFD0100	This is an internal error indicating a failure to create a security descriptor the object. This error may be logged to the product's trace file.

Message	Value (hex)	Description/Action
ABAT-E-INSTSKIPPED, instance skipped due to active instance	CFFD01EC	This audit message indicates that a job instance could not be executed because the template's property did not allow multiple concurrent copies of the job to run simultaneously.
ABAT-E-INSTWAITFAIL, active instance wait failed	CFFD01ED	This audit message indicates that a job instance could not be executed because the "If Job Active… Wait" property was enabled and the timeout period expired.
ABAT-E-INSUFFPRV, Insufficient privilege to perform Batch Queue Management operation	CFFD00FA	You attempted to perform a queue or job operation for which you lack sufficient access permissions.
ABAT-E-INVCALCONFIG, An invalid Calendar configuration was made. Please make sure that all the calendar will allow a job to run at least once	CFFD01FF	You have attempted to create a job and schedule combination in which the job will never run. For example, scheduling a job on December 25th and then assigning a Calendar with Christmas as a holiday.
ABAT-E-INVHISTPROP, Job save history duration must be greater or equal to the log file retension period	CFFD01C6	Self-explanatory.
ABAT-E-INVOPER, An invalid operation was specified	CFFD00DD	Either: an invalid context was specified on an enumeration, or an invalid operation was specified based on a job current state. For example, attempting to release a job which isn't held.
ABAT-E-INVOPNCLSTIME, Invalid queue OPEN time range or CLOSE time range	CFFD0024	The time you entered was not in 24-hour time format. 00:00 through 23:59 is the legal range.
ABAT-E-INVOSPRIORITY, Invalid operating system priority	CFFD0040	An illegal O/S priority value has been specified as part of a job.
ABAT-E-INVOSQPRIORITY, Invalid operating system scheduling priority	CFFD0020	You specified an illegal O/S priority value as the Job default with a queue.
ABAT-E-INVREGVAL, Invalid registry key value, %1!s!	CFFD0009	Data associated with a Registry Key is of an incompatible type. For example, the datatype should be REG_DWORD and the value was found to be REG_SZ; the value itself is not legal.
ABAT-E-INVSTRTSTPTIME, Invalid queue START time range or STOP time range	CFFD0023	The time you entered was not in 24-hour time format. 00:00 through 23:59

Message	Value (hex)	Description/Action
		is the legal range.
ABAT-E-ITEMALREADYEXISTS, The item being added to the collection already exists	CFFD019D	This general error indicates that you are attempting to add an object to a collection and the object already exists. For example, trying to add the same account and access to a security collection.
ABAT-E-JOBABORT, Job aborted during execution	CFFD00C9	This is the exit status code of a job that has been aborted.
ABAT-E-JOBHISTORYERR, History information for the requested job could not be found	CFFD01A9	This is an internal error indicating that the Job's History information could not be retrieved.
ABAT-E-JOBSCHEDFAIL, Job Scheduler failed to start	CFFD019A	This is the exit status code of a Job Scheduler Service which could not be started. Check the trace file or NT Application Event Log for more information.
ABAT-E-JOBSONQUEUE, Jobs exist on queue	CFFD00C4	An attempt to delete a queue failed due to jobs currently on queue. You can indicate that a 'flush' operation first be performed prior to the actual queue delete.
ABAT-E-JSSCONNECTFAIL, Failure to connect to Job Scheduler	CFFD0006	The client was unable to connect to the Job Scheduler machine. Please check the machine name or IP address and try again; the machine may not be on the network.
ABAT-E-KEY_VALUE_NOT_FOUND, Key not found	CFFD006F	This error/warning code indicates that a search for a specific object failed to be found in the database.
ABAT-E-LICENSEXP, license has expired	CFFD0161	The evaluation serial number has expired. Please contact ASCI or your distributor for an extension.
ABAT-E-LOCALONLYLIC, This Personal Edition license is only permitted to communicate with a single Execution Agent on the Local Machine.	CFFD01DA	When using the Personal Edition license you are restricted to a Workstation/Professional NT system and all connections are local to that system.
ABAT-E-LOGONERR, error logging user onto Execution Machine	CFFD00B5	This error indicates that the user's username and/or password was not proper for logging the user onto the

Message	Value (hex)	Description/Action
		Execution Machine. Check the username and password (may be case sensitive) for the target system. More specific details may be found in the job's log file.
ABAT-E-MAXEAGENTLIC, The maximum number of execution agent licenses has been met	CFFD01D3	This licensing error indicates that the maximum number of Execution Agent licenses has been reached. Count each Execution Machine you are attempting to connect to (regardless of whether the machine is active or the queue is started) and then compare against the value reported by the ABATLIC program.
ABAT-E-MSONLYLIC, This Job Scheduler license only permits connections to Microsoft Execution Agents	CFFD01D4	You are attempting to connect to a non-Microsoft Execution Agent and your license only permits you to connect to Microsoft Execution Agents.
ABAT-E-NOBINDINGFND, Binding handle not available or invalid registry %n protocol specified	CFFD000A	This is an error indicating that a protocol sequence established in the Registry is not available on this machine. The RPC protocol sequence Registry key should be examined since it contains erroneous transports.
ABAT-E-NOMACHINENAME, No machine name has been specified	CFFD0003	A machine name needs to be specified. This error most typically occurs as a requirement for an execution queue or when connecting to a Job Scheduler machine.
ABAT-E-NOMORE, No more items	8FFD0178	This message code, typically returned as part of a context search, indicates that no more objects are available.
ABAT-E-NOPASSWORD, No password specified	CFFD0013	Attempting an advanced connection requires a password; attempting to execute a batch job requires a password (unless Blank Password is enabled). Specify the password and try again. Note: You can let Activebatch remember the password for you.
ABAT-E-NOQUEUENAME, No queue name specified	CFFD0010	No queue name was specified and one was required. Specify a valid queue

Message	Value (hex)	Description/Action
		name and try again.
ABAT-E-NOSUCHQUEUE, specified queue does not exist	CFFD00B0	Self-explanatory.
ABAT-E-NOTCONNECTED, A connection has not been successfully made to a Job Scheduler	CFFD019C	You have attempted to perform an operation that requires a current connection to a Job Scheduler machine. Please make the connection and try again.
ABAT-E-NOTGENERIC, This is not a generic queue, unable to associate execution queues	CFFD00F9	Self-explanatory. Check queue properties.
ABAT-E-OPENCMDFILE, Failure to open command procedure	CFFD0160	The Execution Agent was unable to open the batch job's script file. Please check the main job's file specification and ensure it can be opened from the Execution machine.
ABAT-E-PROPERTYREADONLY, You cannot modify this job because it is set as Read Only	CFFD01B5	Self-explanatory.
ABAT-E-QUEUEALRDYEXISTS, The queue already exists	CFFD0038	An attempt to create a queue of the same name failed since queue names must be unique.
ABAT-E-QUEUENOTFND, Queue not found	CFFD002C	The queue name you specified does not exist. Please check the spelling and try again.
ABAT-E-REGOPENERR, Failure to open registry HKEY_LOCAL_MACHINE subkey %1!s!	CFFD001C	ActiveBatch was unable to open a Registry Key value in HKEY_LOCAL_MACHINE. While you should check Registry security permissions this error typically means that the ActiveBatch Registry keys have been corrupted or otherwise tampered with. You should try executing a product REPAIR through the ADD/REMOVE Control Panel applet.
ABAT-E-REGVALERR, Failure to obtain registry value %1!s!	CFFD001D	ActiveBatch was unable to obtain the specified Registry value-name. Typically this means the ActiveBatch Registry keys and/or value names have been corrupted or otherwise tampered with. You should try executing a

Message	Value (hex)	Description/Action
		product REPAIR through the ADD/REMOVE Control Panel applet.
ABAT-E-REQPROPERTYINV, The required property %1!s! is either absent or invalid	CFFD01A5	Self-explanatory.
ABAT-E-RUNJOBINTERACTIVE, Unable to run the job interactively	CFFD01FC	An error occurred while attempting to run this job interactively. The error is most likely due to a lack of desktop permissions.
ABAT-E-SERVERCOMMUNICATIONERROR, There was an error that occurred while communicating with the server	CFFD01A2	The client interface encountered an error while communicating with the Job Scheduler.
ABAT-E-TAKEOWNERFAIL, attempt to take ownership of specified object failed	CFFD01E4	Your attempt to take ownership of this object failed. Verify that you have "Executive" level permission.
ABAT-E-UNEXPEXCEPTCOM, An unexpected exception has occurred in the COM client	CFFD01DF	Self-explanatory.
ABAT-E-WRKSTNONLYLIC, The Personal Edition license is only permitted to run on a Workstation Platform	CFFD01F9	Self-explanatory.
ABAT-I-ADDBATCHPRIV, The account under which this job is executing has been adjusted to have the "Logon as Batch" right	4FFD01DD	The batch user has been automatically granted the "Logon as a Batch User" right.
ABAT-I-DEPENDSAT, dependency is satisfied	4FFD01AD	This audit message indicates that all job dependencies were met.
ABAT-I-EVTTRIGGERED, job triggered by event	4FFD0181	An event occurred which triggered execution of a job.
ABAT-I-JOBSTART, job %1!s!(%2!s!) started successfully on queue %3!s!	4FFD00B1	Self-explanatory. This message is displayed by the command-line client.
ABAT-I-JSSUP, The ActiveBatch Job Scheduler Service has started	4FFD0036	Self-explanatory. Typically found in the JSS Trace file or NT Application Event Log indicating the Job Scheduler Service has been started.
ABAT-I-MARKDELETE, job marked for deletion	4FFD01B4	Typically a template job is marked for deletion when one or more active instances are still executing.
ABAT-I-NOWBEM, WBEM services will not be available on this system	4FFD0191	Microsoft's WMI Services are either not available or have not been started.

Message	Value (hex)	Description/Action
ABAT-S-SUCCESS, normal successful completion	0FFD0001	Operation completed normally. No action to be taken.
ABAT-W-ACCTLOOKUP, unable to lookup account	8FFD019E	ActiveBatch was unable to lookup the specified accounts SID and validate the account. Check any domain and account name spelling.
ABAT-W-ASSIGNJOBFAIL, unable to assign process to job object	8FFD01B1	The Execution Agent could not create a Windows 2000 Job Object within which to encapsulate the batch job.
ABAT-W-CPYFILERR, error copying file to execution machine	8FFD0184	The Job Scheduler was unable to open the job step(s) script/program file. An additional error code provides more information.
ABAT-W-CREPRCERR, error creating batch process for job %1!s!	8FFD00B6	An error occurred creating the batch job process. More specific details may be found in the job's log file.
ABAT-W-EXCRESTART, maximum restart value exceeded	8FFD01D2	The job could not be restarted as it has reached its maximum restart count.
ABAT-W-GETSDERR, unable to obtain Security Descriptor	8FFD01B8	ActiveBatch was unable to find the object's Security Descriptor.
ABAT-W-JOBCANTEXEC, job cannot execute based on Exclusion List/Calendar	8FFD01E9	This audit message appears if a job was scheduled for execution but could not be executed due to an Exclusion List or Calendar conflict.
ABAT-W-JOBNOTEXEC, job not currently executing	8FFD0168	You cannot restart a job that is not currently executing.
ABAT-W-JSSCONNFAIL, failed to connect to Job Scheduler on %1!s!	8FFD00B8	Self-explanatory. Check that the machine's name or IP address is correct; verify that the machine is up and connected to the proper network; verify the RPCSS or DCE services have been started.
ABAT-W-LOADJOBSFAIL, unable to load job objects dll	8FFD01B3	For Windows 2000 machines, the Execution Agent attempts to load a special DLL which associates Windows Job Objects with ActiveBatch jobs. This warning indicates that Windows 2000 Job Objects will not be used.
ABAT-W-NOCOPY, unable to copy files to execution machine	8FFD0183	Self-explanatory. Please check the job step(s) file specifications. They must be accessible from the client machine.

Message	Value (hex)	Description/Action
ABAT-W-NOQUEUESFOUND, no active execution queues found	8FFD0182	This error can occur when a Generic Queue with Round Robin scheduling determines that no active execution queues are available to dispatch a job.
ABAT-W-NOSUCHALERT, no such alert	8FFD017D	The specified alert could not be found for update or deletion purposes.
ABAT-W-NOSUCHCALENDAR, specified calendar does not exist	8FFD01E0	Self-explanatory.
ABAT-W-NOSUCHCHAR, no such characteristic	8FFD0177	The machine or user characteristic you specified could not be found.
ABAT-W-NOSUCHCONTACT, contact information not found	8FFD017A	The specified user/machine contact information could not be found for update or deletion purposes.
ABAT-W-NOSUCHJOB, specified job does not exist	8FFD00BB	Self-explanatory. The job's ID did not match any present jobs in the system.
ABAT-W-NOSUCHPOLICY, specified policy does not exist	8FFD0198	The specified policy does not exist. Check the policy name spelling.
ABAT-W-NOTNT, operation requires Windows NT	8FFD00AF	The Job Scheduler and Execution Agent services cannot be run on non-Windows 2000/NT systems.
ABAT-W-NOTTEMPLATE, not a template job	8FFD0199	The specified job is not a template. Certain operations can only be performed on a template job, for example, you cannot submit or move a non-template job to the OnDemand queue.
ABAT-W-PROCAFFINITY, unable to set processor affinity	8FFD01BF	A processor mask was specified for this job. However, the mask affinity operation could not be performed. The job will still execute. This can be caused if the number of actual processors is less than that specified or an O/S specific restriction on assigning processors to jobs is violated.
ABAT-W-PROPTYLOCKED, property is locked	8FFD018D	An attempt to modify a property failed because the property was locked and cannot be changed by an ActiveBatch non-Administrator.
ABAT-W-REQPARMABS, required parameter absent	8FFD017E	You didn't specify a schedule ID or the query event string for a Job.

Message	Value (hex)	Description/Action
ABAT-W-RPCEXCEPT, RPC Exception thrown	8FFD01B6	This message, if displayed, would be found in a product trace file.
ABAT-W-SCHEDATTACHED, jobs attached to this schedule	8FFD0167	You cannot delete this schedule since jobs are currently associated. You can disassociate the jobs from this schedule and then delete the schedule.
ABAT-W-TERMJOBFAIL, unable to terminate job object	8FFD01B2	The Execution Agent could not terminate the Windows 2000 Job Object. More information typically follows this error.
ABAT-W-TRIGGERFAIL, failed to trigger/clone job	8FFD0192	A job was triggered was unable to be cloned. An additional message provides more information concerning this condition.
ABAT-W-TRIGGERSDISABLED, ActiveBatch Event Triggers are disabled	8FFD01C3	Unable to issue TRIGGER command as ActiveBatch Triggers are not enabled.
ABAT-W-UNEXPEXCEPT, unexpected exception occurred	8FFD0068	An unexpected exception occurred. Typically this message code is not delivered to user code and is found in the product's trace file depending on the trace level specified.
ABAT-W-UNKJSS, unknown Job Scheduler	8FFD01A6	The Execution Agent was unable to reconnect back to the Job Scheduler machine.
ABAT-W-WBEMCONN, unable to connect to WBEM	8FFD01C1	The WMI software has either not been installed or started.
ABAT-W-WMIQUERYFAIL, WMI Query request failed	8FFD01C2	ActiveBatch encountered an error issuing the Event Query string to WMI
Database Error: Communications Error	CFFD00EC	This error usually indicates that insufficient resources have been allocated for the Database server.
Database Error: Record manager inactive	CFFD0079	This is an initialization error and indicates that the Database server has not been started. Verify that W32MKDE.EXE exists within the ActiveBatch installation directory.

Table 13 - ActiveBatch Messages (Alphabetical)

The hexadecimal values below are subject to change and are documented for reference information only.

Value (hex)	Message	Description/Action
0FFD0001	ABAT-S-SUCCESS, normal successful completion	Operation completed normally. No action to be taken.
4FFD0036	ABAT-I-JSSUP, The ActiveBatch Job Scheduler Service has started	Self-explanatory. Typically found in the JSS Trace file or NT Application Event Log indicating the Job Scheduler Service has been started.
4FFD00B1	ABAT-I-JOBSTART, job %1!s!(%2!s!) started successfully on queue %3!s!	Self-explanatory. This message is displayed by the command-line client.
4FFD0181	ABAT-I-EVTTRIGGERED, job triggered by event	An event occurred which triggered execution of a job.
4FFD0191	ABAT-I-NOWBEM, WBEM services will not be available on this system	Microsoft's WMI Services are either not available or have not been started.
4FFD01AD	ABAT-I-DEPENDSAT, dependency is satisfied	This audit message indicates that all job dependencies were met.
4FFD01B4	ABAT-I-MARKDELETE, job marked for deletion	Typically a template job is marked for deletion when one or more active instances are still executing.
4FFD01DD	ABAT-I-ADDBATCHPRIV, The account under which this job is executing has been adjusted to have the "Logon as Batch" right	The batch user has been automatically granted the "Logon as a Batch User" right.
8FFD0068	ABAT-W-UNEXPEXCEPT, unexpected exception occurred	An unexpected exception occurred. Typically this message code is not delivered to user code and is found in the product's trace file depending on the trace level specified.
8FFD00AF	ABAT-W-NOTNT, operation requires Windows NT	The Job Scheduler and Execution Agent services cannot be run on non-Windows 2000/NT systems.
8FFD00B6	ABAT-W-CREPRCERR, error creating batch process for job %1!s!	An error occurred creating the batch job process. More specific details may be found in the job's log file.
8FFD00B8	ABAT-W-JSSCONNFAIL, failed to connect to Job Scheduler on %1!s!	Self-explanatory. Check that the machine's name or IP address is correct; verify that the machine is up and connected to the proper network; verify the RPCSS or DCE services have been started.

Value (hex)	Message	Description/Action
8FFD00BB	ABAT-W-NOSUCHJOB, specified job does not exist	Self-explanatory. The job's ID did not match any present jobs in the system.
8FFD0167	ABAT-W-SCHEDATTACHED, jobs attached to this schedule	You cannot delete this schedule since jobs are currently associated. You can disassociate the jobs from this schedule and then delete the schedule.
8FFD0168	ABAT-W-JOBNOTEXEC, job not currently executing	You cannot restart a job that is not currently executing.
8FFD0177	ABAT-W-NOSUCHCHAR, no such characteristic	The machine or user characteristic you specified could not be found.
8FFD0178	ABAT-E-NOMORE, No more items	This message code, typically returned as part of a context search, indicates that no more objects are available.
8FFD017A	ABAT-W-NOSUCHCONTACT, contact information not found	The specified user/machine contact information could not be found for update or deletion purposes.
8FFD017D	ABAT-W-NOSUCHALERT, no such alert	The specified alert could not be found for update or deletion purposes.
8FFD017E	ABAT-W-REQPARMABS, required parameter absent	You didn't specify a schedule ID or the query event string for a Job.
8FFD0182	ABAT-W-NOQUEUESFOUND, no active execution queues found	This error can occur when a Generic Queue with Round Robin scheduling determines that no active execution queues are available to dispatch a job.
8FFD0183	ABAT-W-NOCOPY, unable to copy files to execution machine	Self-explanatory. Please check the job step(s) file specifications. They must be accessible from the client machine.
8FFD0184	ABAT-W-CPYFILERR, error copying file to execution machine	The Job Scheduler was unable to open the job step(s) script/program file. An additional error code provides more information.
8FFD018D	ABAT-W-PROPTYLOCKED, property is locked	An attempt to modify a property failed because the property was locked and cannot be changed by an ActiveBatch non-Administrator.
8FFD0192	ABAT-W-TRIGGERFAIL, failed to trigger/clone job	A job was triggered was unable to be cloned. An additional message provides more information concerning this condition.
8FFD0198	ABAT-W-NOSUCHPOLICY, specified policy does not exist	The specified policy does not exist. Check the policy name spelling.

Value (hex)	Message	Description/Action
8FFD0199	ABAT-W-NOTTEMPLATE, not a template job	The specified job is not a template. Certain operations can only be performed on a template job, for example, you cannot submit or move a non-template job to the OnDemand queue.
8FFD019E	ABAT-W-ACCTLOOKUP, unable to lookup account	ActiveBatch was unable to lookup the specified accounts SID and validate the account. Check any domain and account name spelling.
8FFD01A6	ABAT-W-UNKJSS, unknown Job Scheduler	The Execution Agent was unable to reconnect back to the Job Scheduler machine.
8FFD01B1	ABAT-W-ASSIGNJOBFAIL, unable to assign process to job object	The Execution Agent could not create a Windows 2000 Job Object within which to encapsulate the batch job.
8FFD01B2	ABAT-W-TERMJOBFAIL, unable to terminate job object	The Execution Agent could not terminate the Windows 2000 Job Object. More information typically follows this error.
8FFD01B3	ABAT-W-LOADJOBSFAIL, unable to load job objects dll	For Windows 2000 machines, the Execution Agent attempts to load a special DLL which associates Windows Job Objects with ActiveBatch jobs. This warning indicates that Windows 2000 Job Objects will not be used.
8FFD01B6	ABAT-W-RPCEXCEPT, RPC Exception thrown	This message, if displayed, would be found in a product trace file.
8FFD01B8	ABAT-W-GETSDERR, unable to obtain Security Descriptor	ActiveBatch was unable to find the object's Security Descriptor.
8FFD01BF	ABAT-W-PROCAFFINITY, unable to set processor affinity	A processor mask was specified for this job. However, the mask affinity operation could not be performed. The job will still execute. This can be caused if the number of actual processors is less than that specified or an O/S specific restriction on assigning processors to jobs is violated.
8FFD01C1	ABAT-W-WBEMCONN, unable to connect to WBEM	The WMI software has either not been installed or started.

Value (hex)	Message	Description/Action
8FFD01C2	ABAT-W-WMIQUERYFAIL, WMI Query request failed	ActiveBatch encountered an error issuing the Event Query string to WMI
8FFD01C3	ABAT-W-TRIGGERSDISABLED, ActiveBatch Event Triggers are disabled	Unable to issue TRIGGER command as ActiveBatch Triggers are not enabled.
8FFD01D2	ABAT-W-EXCRESTART, maximum restart value exceeded	The job could not be restarted as it has reached its maximum restart count.
8FFD01E0	ABAT-W-NOSUCHCALENDAR, specified calendar does not exist	Self-explanatory.
8FFD01E9	ABAT-W-JOBCANTEXEC, job cannot execute based on Exclusion List/Calendar	This audit message appears if a job was scheduled for execution but could not be executed due to an Exclusion List or Calendar conflict.
CFFD0002	ABAT-E-FAILURE, Failure in Batch Queue Management System operation	A non-specific error occurred. The error is typically either internal or transient. Please try the operation again.
CFFD0003	ABAT-E-NOMACHINENAME, No machine name has been specified	A machine name needs to be specified. This error most typically occurs as a requirement for an execution queue or when connecting to a Job Scheduler machine.
CFFD0005	ABAT-E-CONNECTFAIL, Failure to connect to Server, %1!s!	This is a general non-specific error indicating that the client was unable to connect to a server machine. Please check the machine name or IP address and try again.
CFFD0006	ABAT-E-JSSCONNECTFAIL, Failure to connect to Job Scheduler	The client was unable to connect to the Job Scheduler machine. Please check the machine name or IP address and try again; the machine may not be on the network.
CFFD0007	ABAT-E-EMGRCONNECTFAIL, Failure to connect to Execution Queue Manager	The Job Scheduler was unable to connect to the Execution Agent machine. Please check the machine name or IP address and try again; the machine may not be on the network; the machine may not be running RPC or DCE services.
CFFD0008	ABAT-E-ALLOCFAIL, Failure to allocate memory	This is an internal error usually indicating that insufficient hardware or software resources are available to satisfy a memory request.

Value (hex)	Message	Description/Action
CFFD0009	ABAT-E-INVREGVAL, Invalid registry key value, %1!s!	Data associated with a Registry Key is of an incompatible type. For example, the datatype should be REG_DWORD and the value was found to be REG_SZ; the value itself is not legal.
CFFD000A	ABAT-E-NOBINDINGFND, Binding handle not available or invalid registry %n protocol specified	This is an error indicating that a protocol sequence established in the Registry is not available on this machine. The RPC protocol sequence Registry key should be examined since it contains erroneous transports.
CFFD000B	ABAT-E-BADPARAM, Bad parameter received	This is a general error indicating that a bad parameter was specified. Depending on the context you should specify a legal value or property and try again.
CFFD000D	ABAT-E-INITQFAIL, Failure to initialize a queue	The operation to initialize a queue failed. The queue was not created. A more specific error usually accompanies this error.
CFFD000E	ABAT-E-DISCONFAIL, Failure to disconnect from server	An error occurred disconnecting from a Job Scheduler machine. This is typically not something to be overly concerned about since the error occurs on the disconnect itself.
CFFD0010	ABAT-E-NOQUEUENAME, No queue name specified	No queue name was specified and one was required. Specify a valid queue name and try again.
CFFD0013	ABAT-E-NOPASSWORD, No password specified	Attempting an advanced connection requires a password; attempting to execute a batch job requires a password (unless Blank Password is enabled). Specify the password and try again. Note: You can let Activebatch remember the password for you.
CFFD001C	ABAT-E-REGOPENERR, Failure to open registry HKEY_LOCAL_MACHINE subkey %1!s!	ActiveBatch was unable to open a Registry Key value in HKEY_LOCAL_MACHINE. While you should check Registry security permissions this error typically means that the ActiveBatch Registry keys have

Value (hex)	Message	Description/Action
		been corrupted or otherwise tampered with. You should try executing a product REPAIR through the ADD/REMOVE Control Panel applet.
CFFD001D	ABAT-E-REGVALERR, Failure to obtain registry value %1!s!	ActiveBatch was unable to obtain the specified Registry value-name. Typically this means the ActiveBatch Registry keys and/or value names have been corrupted or otherwise tampered with. You should try executing a product REPAIR through the ADD/REMOVE Control Panel applet.
CFFD0020	ABAT-E-INVOSQPRIORITY, Invalid operating system scheduling priority	You specified an illegal O/S priority value as the Job default with a queue.
CFFD0023	ABAT-E-INVSTRTSTPTIME, Invalid queue START time range or STOP time range	The time you entered was not in 24-hour time format. 00:00 through 23:59 is the legal range.
CFFD0024	ABAT-E-INVOPNCLSTIME, Invalid queue OPEN time range or CLOSE time range	The time you entered was not in 24-hour time format. 00:00 through 23:59 is the legal range.
CFFD002C	ABAT-E-QUEUENOTFND, Queue not found	The queue name you specified does not exist. Please check the spelling and try again.
CFFD0033	ABAT-E-EMGRFAIL, Network connection to the execution machine has been lost	Self-explanatory. Depending on network conditions this problem may resolve itself as the machine is re-discovered.
CFFD0038	ABAT-E-QUEUEALRDYEXISTS, The queue already exists	An attempt to create a queue of the same name failed since queue names must be unique.
CFFD0040	ABAT-E-INVOSPRIORITY, Invalid operating system priority	An illegal O/S priority value has been specified as part of a job.
CFFD006F	ABAT-E-KEY_VALUE_NOT_FOUND, Key not found	This error/warning code indicates that a search for a specific object failed to be found in the database.
CFFD0070	ABAT-E-DUPLICATE_KEY_VALUE, Duplicate key value	This error is returned when an attempt has been made to create an object with a duplicate key (2 objects with the same name).
CFFD0079	Database Error: Record manager inactive	This is an initialization error and indicates that the Database server has

Value (hex)	Message	Description/Action
		not been started. Verify that W32MKDE.EXE exists within the ActiveBatch installation directory.
CFFD00B0	ABAT-E-NOSUCHQUEUE, specified queue does not exist	Self-explanatory.
CFFD00B5	ABAT-E-LOGONERR, error logging user onto Execution Machine	This error indicates that the user's username and/or password was not proper for logging the user onto the Execution Machine. Check the username and password (may be case sensitive) for the target system. More specific details may be found in the job's log file.
CFFD00C4	ABAT-E-JOBSONQUEUE, Jobs exist on queue	An attempt to delete a queue failed due to jobs currently on queue. You can indicate that a 'flush' operation first be performed prior to the actual queue delete.
CFFD00C5	ABAT-E-ABORTENTRY, Entry has been aborted	A job has been marked as 'aborted'.
CFFD00C9	ABAT-E-JOBABORT, Job aborted during execution	This is the exit status code of a job that has been aborted.
CFFD00CF	ABAT-E-IMPERSONERR, Failure to impersonate client	This error can be returned if you attempt to connect to a Job Scheduler while the Scheduler is still in the process of starting. Please wait a moment and try again.
CFFD00DD	ABAT-E-INVOPER, An invalid operation was specified	Either: an invalid context was specified on an enumeration, or an invalid operation was specified based on a job current state. For example, attempting to release a job which isn't held.
CFFD00EC	Database Error: Communications Error	This error usually indicates that insufficient resources have been allocated for the Database server.
CFFD00F9	ABAT-E-NOTGENERIC, This is not a generic queue, unable to associate execution queues	Self-explanatory. Check queue properties.
CFFD00FA	ABAT-E-INSUFFPRV, Insufficient privilege to perform Batch Queue Management operation	You attempted to perform a queue or job operation for which you lack sufficient access permissions.

Value (hex)	Message	Description/Action
CFFD0100	ABAT-E-INITSD, Failure to initialize ABAT Security Descriptor	This is an internal error indicating a failure to create a security descriptor the object. This error may be logged to the product's trace file.
CFFD015E	ABAT-E-ASSOCNOTFND, Failure to determine file association	The Execution Agent was unable to find the job file's extension to program association. The job was not run.
CFFD015F	ABAT-E-CRELOGFILE, Failure to create log file	The Execution Agent was unable to create the batch job's log file. Typically, the log file specification is not valid or proper OR the main job script file specification is also not valid. Please check the log/script file specification and ensure it can be created from the Execution machine.
CFFD0160	ABAT-E-OPENCMDFILE, Failure to open command procedure	The Execution Agent was unable to open the batch job's script file. Please check the main job's file specification and ensure it can be opened from the Execution machine.
CFFD0161	ABAT-E-LICENSEXP, license has expired	The evaluation serial number has expired. Please contact ASCI or your distributor for an extension.
CFFD019A	ABAT-E-JOBSCHEDFAIL, Job Scheduler failed to start	This is the exit status code of a Job Scheduler Service which could not be started. Check the trace file or NT Application Event Log for more information.
CFFD019C	ABAT-E-NOTCONNECTED, A connection has not been successfully made to a Job Scheduler	You have attempted to perform an operation that requires a current connection to a Job Scheduler machine. Please make the connection and try again.
CFFD019D	ABAT-E-ITEMALREADYEXISTS, The item being added to the collection already exists	This general error indicates that you are attempting to add an object to a collection and the object already exists. For example, trying to add the same account and access to a security collection.
CFFD01A2	ABAT-E-SERVERCOMMUNICATIONERROR, There	The client interface encountered an error while communicating with the Job

Value (hex)	Message	Description/Action
	was an error that occurred while communicating with the server	Scheduler.
CFFD01A5	ABAT-E-REQPROPERTYINV, The required property %1!s! is either absent or invalid	Self-explanatory.
CFFD01A7	ABAT-E-COPYFILEERROR, An error occurred while copying the file '%1!s!' to the Job Scheduler because the file cannot be found or there was an error opening/reading the file.	This message can occur if the "Copy File" feature is enabled for a Main, Pre or Post file. As the message indicates an error occurred while attempting to copy the file to the Job Scheduler.
CFFD01A8	ABAT-E-FILESIZEERROR, The file being copied is larger than the system policy allows for	The ActiveBatch Administrator has prevented your file from being copied to the Execution Machine through the System policy CopyScriptMax
CFFD01A9	ABAT-E-JOBHISTORYERR, History information for the requested job could not be found	This is an internal error indicating that the Job's History information could not be retrieved.
CFFD01AE	ABAT-E-DEPENDFAIL, dependency failure	The Batch Job failed because a job dependency could not be satisfied.
CFFD01B5	ABAT-E-PROPERTYREADONLY, You cannot modify this job because it is set as Read Only	Self-explanatory.
CFFD01C4	ABAT-E-EXCRUNTIME, job aborted exceeding expected run time	The Execution Agent has set this job's exit status code because of an overrun for elapsed run time and has aborted the job.
CFFD01C5	ABAT-E-EXCCPUTIME, job aborted exceeding expected cpu time	The Execution Agent has set this job's exit status code because of an overrun for CPU time and has aborted the job.
CFFD01C6	ABAT-E-INVHISTPROP, Job save history duration must be greater or equal to the log file retension period	Self-explanatory.
CFFD01CA	ABAT-E-BATCHSTARTERR, unable to launch Batch Start job	A job completed and had a Batch Start (Success or Failure) action associated with it. That action could not be performed. Additional information accompanies this message indicating more specifically what the problem was.
CFFD01D1	ABAT-E-EXECABORT, job aborted due to Execution Agent failure	A job was aborted when an Execution Agent/machine failed. The job was not marked as being restartable.

Value (hex)	Message	Description/Action
CFFD01D3	ABAT-E-MAXEAGENTLIC, The maximum number of execution agent licenses has been met	This licensing error indicates that the maximum number of Execution Agent licenses has been reached. Count each Execution Machine you are attempting to connect to (regardless of whether the machine is active or the queue is started) and then compare against the value reported by the ABATLIC program.
CFFD01D4	ABAT-E-MSONLYLIC, This Job Scheduler license only permits connections to Microsoft Execution Agents	You are attempting to connect to a non-Microsoft Execution Agent and your license only permits you to connect to Microsoft Execution Agents.
CFFD01D7	ABAT-E-DUPLICFND, A duplicate license was found. Please ensure that all licenses are valid and unique.	This serious error usually indicates some level of license tampering. Please run the ABATLIC program to obtain more information.
CFFD01DA	ABAT-E-LOCALONLYLIC, This Personal Edition license is only permitted to communicate with a single Execution Agent on the Local Machine.	When using the Personal Edition license you are restricted to a Workstation/Professional NT system and all connections are local to that system.
CFFD01DF	ABAT-E-UNEXPEXCEPTCOM, An unexpected exception has occurred in the COM client	Self-explanatory.
CFFD01E4	ABAT-E-TAKEOWNERFAIL, attempt to take ownership of specified object failed	Your attempt to take ownership of this object failed. Verify that you have “Executive” level permission.
CFFD01E5	ABAT-E-GETOWNER, unable to obtain object owner	ActiveBatch is unable to retrieve the owner of the object. This error may appear if the domain of the owner’s account is not available.
CFFD01EA	ABAT-E-DBPURGEFAIL, An error occurred while running the DBPurge process. Please examine the NT Event Log/DBPurge Log for further details	Self-explanatory. This error message concerns the running of the DBPurge facility to prune unneeded jobs from the ActiveBatch system.
CFFD01EB	ABAT-E-AUDPURGEFAIL, Failed to purge Audit ID %1!d!	This DBPurge message can appear in the NT Application Event Log and indicates that DBPurge was unable to remove old audit messages from a specific job.

Value (hex)	Message	Description/Action
CFFD01EC	ABAT-E-INSTSKIPPED, instance skipped due to active instance	This audit message indicates that a job instance could not be executed because the template's property did not allow multiple concurrent copies of the job to run simultaneously.
CFFD01ED	ABAT-E-INSTWAITFAIL, active instance wait failed	This audit message indicates that a job instance could not be executed because the "If Job Active… Wait" property was enabled and the timeout period expired.
CFFD01F9	ABAT-E-WRKSTNONLYLIC, The Personal Edition license is only permitted to run on a Workstation Platform	Self-explanatory.
CFFD01FC	ABAT-E-RUNJOBINTERACTIVE, Unable to run the job interactively	An error occurred while attempting to run this job interactively. The error is most likely due to a lack of desktop permissions.
CFFD01FF	ABAT-E-INVCALCONFIG, An invalid Calendar configuration was made. Please make sure that all the calendar will allow a job to run at least once	You have attempted to create a job and schedule combination in which the job will never run. For example, scheduling a job on December 25th and then assigning a Calendar with Christmas as a holiday.

Table 14 - ActiveBatch Messages (Numeric)

12 INDEX

A

ABATLIC 26, 27, 314, 318, 333
Absolute Time 268
ActiveBatch
  icons 84
  Introduction 7
  Registration 11, 12, 17
  Starting 30
  Stopping 32
  Support 10, 11, 12, 28
  Warranty 5, 12
ActiveBatch Admin Application8, 14, 46, 51, 54, 55, 56, 74, 77, 78, 79, 84, 87, 97, 98, 136, 140, 193, 198, 200, 259, 260, 261, 267
ActiveBatch Administrators ... 130, 246, 264, 265
ASCI_ABATLOG 25, 161

B

BQMS 2, 37, 38
Business Day 48, 106, 139, 221, 223

C

Calendar 87, 106, 221, 264
Client8, 9, 10, 14, 15, 16, 19, 25, 97, 113, 139, 160, 188, 220, 246, 262, 276, 285, 311
CLOSE/QUEUE 271
Combination Time 270
Connect 54, 55, 79, 87, 98, 99, 251
CopyFolder 28
Creating a Generic Queue 118
Creating an Execution Queue 57, 109

D

DataPath 28
DBPurge 36, 152, 259, 313, 314, 333
DELETE/JOB 272
DELETE/QUEUE 273
Delta Time 269

E

Error Messages 313, 323, 334
Evaluation 1, 2, 16, 17
Event Triggers 140, 181, 261, 323, 327
ExecAutoRight 29, 130
ExecCopyFolder 29
ExecLogFiles 25, 29
ExecLogFolder 29
Execution Agent8, 9, 10, 13, 14, 16, 22, 25, 26, 27, 29, 30, 31, 32, 39, 46, 48, 59, 81, 84, 113, 130, 148, 166, 211, 239, 253, 254, 313, 314, 315, 317, 318, 319, 321, 322, 323, 324, 326, 327, 331, 332, 333
Execution Machine9, 29, 47, 63, 112, 135, 148, 188, 205, 315, 317, 318, 330, 332, 333
Execution Queue9, 48, 57, 58, 59, 60, 62, 68, 80, 81, 84, 85, 109, 112, 113, 116, 118, 119, 120, 123, 128, 130, 137, 141, 142, 152, 165, 166, 167, 182, 197, 200, 204, 205, 206, 207, 208, 209, 228, 238, 278, 287, 299, 314, 327

F

Failure Batch Start 140, 153, 185
FLUSH/QUEUE 274

G

Generic Queue
  Scheduling Algorithm120, 137, 206, 208, 258, 260

H

HOLD 275, 277, 281, 284, 304
Hold Job 111, 140, 152, 235, 261
Holiday 106, 221, 222, 223

Hot Keys...87

I

INITIALIZE Queue.......................................276
Installation.................... 13, 16, 25, 161, 248, 259
Introductory Assistance Wizard **52**

J

Job
  Alerts.................. 140, 149, 150, 187, 188, 236
  Audit Trail..160, 168
  Central Manage Log File....................150, 241
    Retention 150
  Date/Time Trigger.............................139, 261
    Immediate................................68, 139, 212
    Interval 94, 139, 179, 212, 213
  Environmental variables............................146
  File Name 135, 136, 150
  History..159
    Retention 152, 262
  If Job Active 135, 147, 262, 316, 334
  Interactive..148
  Log File 161, 241
  Maximum Restarts 152
  Monitor CPU Time....................................155
  Monitor Elapsed Time...............................155
  Notifications 153, 186, 236
  Parameters135, 158, 188, 197, 201, 280, 303, 309
  Pre/Post Steps....................................146, 156
  Processor Mask (SMP)..............103, 148, 261
  Queue Selection..................137, 195, 209, 262
  Restart 152
  Simulation 156
  Submission Queue.............................137, 161
  Success Code Rule 64, 153, 211, 229, 262
  Terminate all child processes 152
  Working Directory 113, 137
Job Execution Limit 113, 205
Job Scheduler8, 13, 14, 15, 17, 20, 24, 26, 28, 30, 31, 32, 36, 46, 47, 54, 55, 57, 58, 78, 79, 80, 81, 82, 86, 87, 90, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 109, 112, 113, 117, 135, 139, 146, 147, 148, 163, 169, 180, 181, 182, 188, 195, 203, 220, 224, 228, 237, 239, 253, 254, 259, 262, 267, 271, 272, 273, 274, 275, 277, 281, 282, 284, 286, 288, 289, 290, 291, 292, 293, 294, 295, 296, 298, 300, 301, 304, 305, 307, 308, 309, 314, 315, 317, 318, 319, 320, 321, 322, 323, 324, 325, 326, 327, 328, 330, 331, 333
JobDefinition 130

L

License 3, 6
  Evaluation...1
  Standard..1, 3
Log View...56, 79, 86

M

Machine Characteristics .113, 141, 165, 166, 207
MMC8, 14, 15, 19, 28, 32, 36, 47, 51, 57, 102, 115, 246, 247, 248, 250, 251, 252, 254, 255, 256, 257, 258, 259, 260, 261, 263, 264, 265, 267
  Policies 257
    Job 103, 111, 157, 252, 261, 262, 267
    Queue 102, 114, 252, 260
    System 36, 252, 259
  Registry Entries 255
  Services 254
MMC Snap-In................................247, 248, 249
MODIFY/JOB 279, 284
MODIFY/QUEUE.................................285, 287
MOF 25

N

Navigation Bar .54, 60, 79, 80, 81, 82, 83, 84, 86
Navigation View.................54, 67, 72, 79, 84, 86
Neighborhood View 79
Non-Microsoft Execution Agents.....................39
NTMSG....................64, 153, 212, 262, 283, 307

O

O/S Priority..111, 235

Objects 47, 321, 326
  Event Object 48
  Job Object 47, 321, 323, 326
  Queue Object 47
  Schedule and Calendar Objects 48
OnDemand48, 57, 80, 122, 188, 196, 197, 200, 201, 322, 326
OPEN/QUEUE 288
OpenVMS9, 14, 63, 113, 146, 152, 153, 211, 212, 284, 307
Operating Times 111

P

PAUSE 289
Product Editions 3, 4, 8, 9, 10
  Advanced Server 9, 10, 26
  Enterprise Cross-Platform Server 9
  Personal 9, 10, 26, 317, 320, 333, 334
  Small Business Server 9, 10, 26

Q

Queue
  Counters 121
Queue Name 110
Queue Priority 111, 147, 235, 260, 299
Queue Security 115, 118
Quick Tour
  Scheduled Job Submission 68
  Simple Job Submission 62

R

RELEASE 274, 290, 291
Reporting
  Export 89, 92
Reporting View 86
REQUEUE 291
RESTART 146, 202, 238, 283, 292, 306
RESUME 293
Runbook View 67, 73, 79, 86, 87, 88, 94, 97, 195

S

Schedule38, 48, 69, 70, 87, 106, 108, 139, 169, 171, 216, 264
  Associate 175
  Copying 174
  Create 170, 171
  Delete 174
  Disassociate 175
  Edit 174
  Security 108, 176
serial number 1, 4, 11, 12, 16, 18, 26, 317, 331
SET BATCH 277, 286, 294, 306
SHOW JOB 296, 297
SHOW QUEUE 298, 299
SMTP 28, 29, 31, 186, 188, 236
SMTPFrom 28, 31
SMTPPort 29, 31
SMTPServer 29, 31
START/QUEUE 300
STOP/QUEUE 301
SUBMIT38, 267, 277, 281, 286, 294, 295, 302, 304, 307
Submitting a Job 130, 133
Success Batch Start 153, 233
Success Code Rule 64, 153, 211, 229, 262
Sun Solaris 9, 14
SYNCHRONIZE 308

T

Tool Bar 54, 86
TRIGGER 279, 280, 302, 303, 309, 323, 327
Tru64 UNIX 9, 205

U

User Characteristics 113, 142, 167, 205, 206, 207

W

WMI15, 20, 25, 48, 86, 97, 101, 181, 182, 183, 184, 223, 224, 225, 226, 227, 228, 320, 323, 324, 326, 327
  refresh 86
Working Directory 113, 137

X

XLNT 2, 37, 64, 153, 267, 297, 299