Tivoli® software

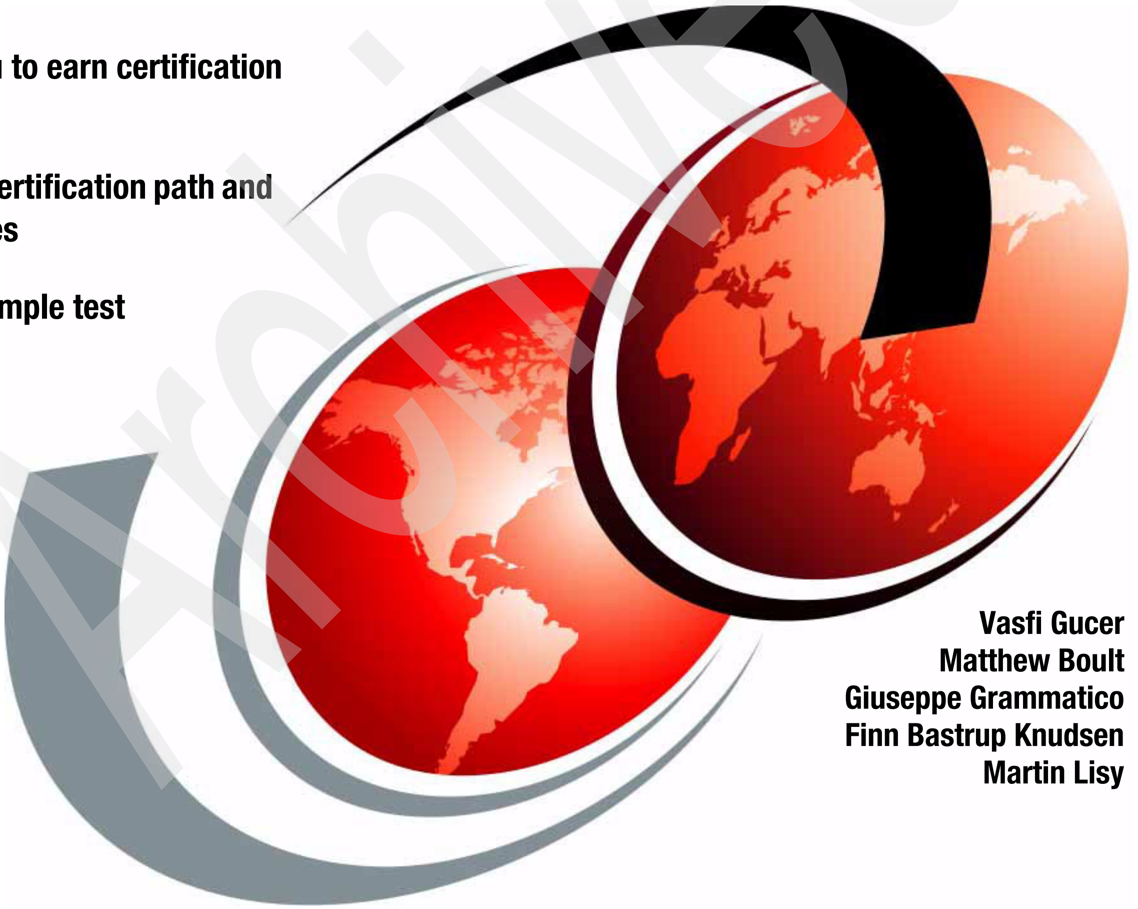
IBM

Certification Guide Series: IBM Tivoli Workload Scheduler V8.4

Enables you to earn certification

Describes certification path and prerequisites

Provides sample test questions



Vasfi Gucer
Matthew Boult
Giuseppe Grammatico
Finn Bastrup Knudsen
Martin Lisy

Redbooks

ibm.com/redbooks

International Technical Support Organization

Certification Guide Series: IBM Tivoli Workload Scheduler V8.4

August 2008

SG24-7628-00

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

First Edition (August 2008)

This edition applies to IBM Tivoli Workload Scheduler Version 8, Release 4.

Contents

Figures

Tables

Examples

Notices

Trademarks

AIX®	Maestro™	Redbooks (logo) ®
BetaWorks™	NetView®	Tivoli Enterprise Console®
DB2®	OS/400®	Tivoli®
i5/OS®	POWER™	WebSphere®
IBM®	Redbooks®	z/OS®

Preface

This IBM® Redbooks® publication is a study guide for IBM Tivoli® Workload Scheduler Version 8.4 and is aimed at readers who want to obtain IBM Professional Certification for this product.

The IBM Tivoli Workload Scheduler Version 8.4 Certification, offered through the Professional Certification Program from IBM, is designed to validate the required skills for technical professionals who implement the IBM Tivoli Workload Scheduler Version 8.4 product.

This book provides a combination of theory and practical experience required for a general understanding of the product. It also provides sample questions that help you evaluate your personal progress and provide familiarity with the types of questions that you encounter in the exam.

This publication does not replace practical experience, nor is it designed to be a standalone guide. Instead, when combined with education activities and experience, this book is an effective tool that can provide invaluable preparation for the exam.

For your convenience, we structure the chapters based on the sections of the IBM Tivoli Workload Scheduler V8.4 Implementation Certification test, such as planning, installation, and configuration. Each topic corresponds to one section of the exam and enables you to prepare for that section.

The team that wrote this book

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

Vasfi Gucer is a Project Leader at the ITSO, Austin Center. He worked for IBM Turkey for 10 years and has been with the ITSO Austin Center since January 1999. He has more than 12 years of experience in the areas of systems management, networking hardware, and software on mainframe and distributed platforms. He has worked on various Tivoli customer projects as a Systems Architect and Technical Project Manager. He writes extensively and teaches IBM classes worldwide on Tivoli software. He is also an IBM Certified Senior IT Specialist.

Matthew Boult is a Product Introduction Specialist in the IBM BetaWorks™ organization. He is based in Hursley in the U.K. and runs worldwide beta programs for Tivoli products, generating feedback to development, creating reference accounts, and promoting sales enablement through early education and the production of intellectual capital. He has more than 25 years of experience in the IT industry and has been working with Tivoli since the acquisition of the company by IBM, initially providing post-sales technical support and later designing and implementing solutions for outsourced customers. He has run beta programs for Tivoli Workload Scheduler since the release of Version 8.2 in 2003.

Giuseppe Grammatico is a Software Support Specialist working for Italy IMT in the Tivoli Customer Support team within IBM Global Technology Services. He has been working for IBM since 2002 and has strong skills in several Tivoli product suites. During these years he has been involved in several projects implementing Tivoli solutions for IBM clients in Italy - such solutions as Tivoli Workload Scheduler, Tivoli Management Framework, Tivoli Storage Manager, Tivoli Directory Servers, and Tivoli Identity Manager. In April 2006, Giuseppe was awarded a Professional Certification from IBM for Tivoli Workload Scheduler V8.2.

Finn Bastrup Knudsen is an Senior IT Specialist in Integrated Technology Services (ITS) in IBM Global Services in Copenhagen, Denmark. He has worked at IBM for 19 years. He has 9 years of experience working with IBM Tivoli Workload Scheduler and 17 years of experience with IBM Tivoli Workload Scheduler for z/OS®. Finn primarily provides consultation and services at customer sites, as well as IBM Tivoli Workload Scheduler for z/OS and IBM Tivoli Workload Scheduler training. He is a certified Tivoli Instructor in IBM Tivoli Workload Scheduler and IBM Tivoli Workload Scheduler for z/OS.

Martin Lisy has been working for IBM since 1995. He is a Systems Management Specialist in the area of the Tivoli Enterprise Management Solutions. Martin has worked on various Tivoli customer projects as the Solution Designer and Implementor. His area of expertise is Tivoli Workload Scheduler, for which he has created various scripting functions that enhance the product capabilities. Martin is an IBM Certified Deployment Professional for Tivoli Workload Scheduler V8.2 and V8.3 and an IBM Certified Deployment Professional for Tivoli Enterprise Console® V3.9. He works with IBM enablement teams in the area of workload management, where he focuses on requirements definitions for new releases of Tivoli Workload Scheduler and Tivoli Dynamic Workload Broker. He also participates in Tivoli certification tests development. Martin holds an engineering degree in computer science from VSB-Technical University of Ostrava.

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

Arzu Gucer
Nancy Crumpton
International Technical Support Organization, Austin Center

Arcangelo Di Balsamo, Liliana Pasceri
IBM Italy

Kimberly Arceneaux, Jackie Biggs, Warren Gill
IBM U.S.

Michael A Lowry
IBM Sweden

Bohuslav Bezecny
CEZData, s.r.o.

Become a published author

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

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

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

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

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

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

  ibm.com/redbooks

- Send your comments in an e-mail to:

  redbooks@us.ibm.com

- Mail your comments to:

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

1

Certification overview

This chapter provides an overview of the skill required to become an IBM certified Advanced Technical Expert.We designed the following sections to provide a comprehensive review of specific topics that provide essential information for obtaining the certification:

- “IBM Professional Certification Program” on page 2
- “Test 435: Workload Scheduler V8.4 Implementation” on page 7
- “Recommended resources for study” on page 8

1.1 IBM Professional Certification Program

Having the right skills for the job is critical in the growing global marketplace. IBM Professional Certification, designed to validate skill and proficiency in the latest IBM solution and product technology, can help provide that competitive edge. The IBM Professional Certification Program Web site is available at:

```
http://www.ibm.com/certify/index.shtml
```

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

In addition to demonstrating your skill and proficiency in the latest IBM technology and solutions, professional certification can help you excel at your job by giving you and your employer confidence that your skills have been tested. You can deliver higher levels of service and technical expertise than non-certified employees and move on a faster career track. Professional certification puts your career in your control.

The certification requirements are difficult but not overwhelming. Certification is a rigorous process that differentiates you from everyone else.

The mission of the IBM Professional Certification Program is to:

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

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

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

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

1.1.1 Benefits of certification

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

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

- For employees:
  - Promotes recognition as an IBM certified professional
  - Creates advantages in interviews
  - Assists in salary increases, corporate advancement, or both
  - Increases self-esteem
  - Provides continuing professional benefits
- For employers:
  - Measures the effectiveness of training
  - Reduces course redundancy and unnecessary expenses
  - Provides objective benchmarks for validating skills
  - Facilitates long-range planning
  - Helps to manage professional development
  - Aids as a hiring tool
  - Contributes to competitive advantage
  - Increases productivity
  - Increases morale and loyalty
- For IBM Business Partners and consultants:
  - Provides independent validation of technical skills
  - Creates competitive advantage and business opportunities
  - Enhances prestige of the team
  - Contributes to meeting IBM requirements for various IBM Business Partner programs

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

- Industry recognition

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

- Program credentials

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

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

- Ongoing technical vitality

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

1.1.2 Tivoli Software Professional Certification

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

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

Whether you are a Tivoli client, partner, or technical professional wanting to put your career on the fast track, you can start on the road to becoming a Tivoli Certified Professional today.

Benefits of Tivoli certification

This section describes the benefits of IBM Tivoli certification. Tivoli certification provides the following benefits:

- For the individual:
  - IBM Certified certificate and use of logos on business cards

Note: Certificates are sent by e-mail. However, a paper copy of the certificate along with a laminated wallet card can also be requested by sending an e-mail to `mailto:certify@us.ibm.com`.

- Recognition of your technical skills by your peers and management
- Enhanced career opportunities
- Focus for your professional development

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

Certification checklist

This section provides a certification checklist. Follow these steps to pursue certification:

1. Select the certification that you want to pursue.
2. Determine which test or tests are required by reading the certification role description.
3. Prepare for the test, using the following resources provided:
   - Test objectives
   - Recommended educational resources
   - Sample assessment test
   - Other reference materials
   - List of opportunities for gaining experience

   Note: These resources are available from each certification description page, as well as from the test information page.

4. Register to take a test by contacting one of our worldwide testing vendors:
   - Thomson Prometric
   - Pearson Virtual University Enterprises (VUE)

   Note: When providing your name and address to the testing vendor, be sure to specify your name exactly as you want it to appear on your certificate.

5. Take the test. Be sure to keep the Examination Score Report provided upon test completion as your record of taking the test.

   Note: After taking a test, your test results and demographic data (including name, address, e-mail, and phone number) are sent from the testing vendor to IBM for processing (allow two to three days for transmittal and processing). After all the tests required for a certification are passed and received by IBM, your certificate is issued.

6. Repeat steps 3 on page 5 through 5 until all required tests are successfully completed for the desired certification role. If you must meet additional requirements (such as an “other vendor” certification or exam), follow the instructions on the certification description page to submit these requirements to IBM.
7. After you complete your certification requirements, you are sent an e-mail asking you to accept the terms of the IBM Certification Agreement before receiving the certificate.
8. Upon acceptance of the terms of the IBM Certification Agreement, an e-mail is sent containing the following electronic deliverables:
   - A Certification certificate in PDF format, which can be printed in either color or black and white
   - A set of graphic files of the IBM Professional Certification mark associated with the certification achieved
   - Guidelines for the use of the IBM Professional Certification mark
9. To avoid unnecessary delay in receiving your certificate, ensure that your current e-mail is on file by maintaining an up-to-date profile. If you do not have an e-mail address on file, your certificate is sent through postal mail.

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

Note: IBM reserves the right to change or delete any portion of the program, including the terms and conditions of the IBM Certification Agreement, at any time without notice. Some certification roles offered through the IBM Professional Certification Program require recertification.

1.2 Test 435: Workload Scheduler V8.4 Implementation

This section describes the process of obtaining Tivoli Workload Scheduler V8.4 certification. We can categorize the certification process as follows:

- Job role description and target audience:

  An IBM Certified Deployment Professional - IBM Tivoli Workload Scheduler V8.4 is a technical professional responsible for planning, installation, configuration, operations, administration, and maintenance of an IBM Tivoli Workload Scheduler V8.4 solution. This individual is expected to perform these tasks with limited assistance from peers, product documentation, and support resources.

  To attain the IBM Certified Deployment Professional - IBM Tivoli Workload Scheduler V8.4 certification, candidates must pass one test.

- The required prerequisites are as follows:
  - Strong working knowledge of IBM Tivoli Workload Scheduler V8.4 infrastructure components
  - Multiplatform system skills (such as UNIX® and Microsoft® Windows®), including working knowledge of accounts, systems, disks, and network administration.
  - Experience with Tivoli Workload Scheduler for Applications (for example, SAP® applications and PeopleSoft® enterprise applications)
  - Knowledge of Tivoli Workload Scheduler networking (TCP/IP, DNS, firewalls)
  - Ability to perform basic diagnosis and isolate issues with the various components of Tivoli Workload Scheduler V8.4 as well as DB2® and Oracle® (or supported relational databases) and the WebSphere® Application Server.
  - Project plan development (for example, architecture, solution design, deployment environment analysis)
  - Tivoli Workload Scheduler V8.4 deployment with limited assistance
  - Knowledge of batch scheduling concepts in a production environment

  - Knowledge transfer of basic product skills such as administration, scheduling, and operations
  - Experience with IBM Tivoli Dynamic Workload Console
  - Knowledge of event-based scheduling
- To be certified you must select Test 435: IBM Tivoli Workload Scheduler V8.4 Implementation:
  - Approximate number of questions: 79
  - Duration in minutes: 105
  - Format: Multiple choice
  - Required passing score: 70%

1.2.1 Test 435 objectives

For the most updated objectives of the IBM Tivoli Workload Scheduler V8.4 Implementation certification test, refer to the following link:

http://www-03.ibm.com/certify/tests/obj435.shtml

1.3 Recommended resources for study

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

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

Note: Course offerings are continuously being added and updated. If you do not see the courses listed in your location, contact one of the previously listed delivery management teams.

1.3.1 Courses

Course names and course numbers vary depending on the education delivery arm used in each geographical location.

Refer to the following link for a list of courses related to IBM Tivoli Workload Scheduler V8.4:

`http://www-03.ibm.com/certify/tests/edu435.shtml`

1.3.2 Publications

Before taking Test 435, IBM Tivoli Workload Scheduler V8.4 Implementation, we recommend that you review IBM Tivoli Workload Scheduler V8.4 guides and IBM Redbooks publications.

For online publications of IBM Tivoli Workload Scheduler V8.4, refer to the following link:

`http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?toc=/com.ibm.tivoli.itws.doc/toc.xml`

IBM Tivoli Workload Scheduler V8.4 Redbooks publications

You can refer to the IBM Redbooks publication, *Deployment Guide Series: IBM Tivoli Workload Scheduler V8.4 and IBM Tivoli Dynamic Workload Broker V1.2*, SG24-75288, as a study resource. This book covers different scheduling scenarios with Tivoli Workload Scheduler V8.4 and Tivoli Dynamic Workload Broker V1.2 in distributed environments. It also discusses best practices for designing and implementing a scheduling solution for small, medium, and enterprise accounts.

2

Planning

This chapter describes the Tivoli Workload Scheduler architecture and the required hardware and software for installing various Tivoli Workload Scheduler components. The following topics are covered:

2.1 Overview

Tivoli Workload Scheduler is the centralized enterprise scheduling application from IBM that integrates scheduling across all systems in the enterprise. It can run on almost every platform in common use today, including most varieties of UNIX, Linux®, Microsoft Windows, i5/OS®, and z/OS.

Tivoli Workload Scheduler is fault tolerant: In the event of a power loss or network outage, it simply carries on where it left off after the affected resource is back up and running again. Using Tivoli Workload Scheduler, you can set up complex dependencies on your jobs so that the workload schema reflects the way a business works. A Tivoli Workload Scheduler job is never allowed to run until all of its dependencies across the whole network have been met. In addition, by using Tivoli Workload Scheduler for Applications, it is possible to integrate Tivoli Workload Scheduler with other enterprise applications such as Oracle and SAP - for example, enabling you to schedule a job on your SAP system after both a UNIX job and a mainframe job have completed.

2.2 Tivoli Workload Scheduler architecture

Tivoli Workload Scheduler helps you plan every phase of production. During the processing day, the Tivoli Workload Scheduler production control programs manage the production environment and automate most operator activities. Tivoli Workload Scheduler prepares jobs for execution, resolves interdependencies, and launches and tracks each job. Because jobs start running as soon as their dependencies are satisfied, idle time is minimized, and throughput is improved. Jobs never run out of sequence. If a job ends in error, Tivoli Workload Scheduler handles the recovery process with little or no operator intervention.

Tivoli Workload Scheduler is composed of the following several components:

- Tivoli Workload Scheduler engine

  The Tivoli Workload Scheduler engine is installed on every non-mainframe workstation in the scheduling network (for example, UNIX, Microsoft Windows, and OS/400® computers). When the engine is installed on a workstation, it can be configured to play a specific role in the scheduling network. For example, the engine can be configured to be a master domain manager, a domain manager, or a fault-tolerant agent (FTA). In an ordinary Tivoli Workload Scheduler network, a single master domain manager is at the top of the network.

- Tivoli Workload Scheduler database

  The Tivoli Workload Scheduler database stores all of the scheduling objects including jobs and job streams, and all of the dependencies defined on them. By default, Tivoli Workload Scheduler uses an integrated DB/2 database to store this data; however, the data can instead be stored in an previously installed Oracle database if desired.

- Tivoli Workload Scheduler connector

  An integrated version of IBM WebSphere Application Server Express provides communication between the Tivoli Workload Scheduler client interfaces (the command line, Job Scheduling Console, and Tivoli Dynamic Workload Console) and the back-end Tivoli Workload Scheduler engine and database.

- Job Scheduling Console (JSC)

  Job Scheduling Console is the Java™-based graphical user interface for the Tivoli Workload Scheduler suite. The Job Scheduling Console runs on any machine from which you want to manage Tivoli Workload Scheduler plan and database objects. It provides, through the Tivoli Workload Scheduler connector, the functions of the command-line programs conman and composer. The JSC can be installed on a desktop workstation or mobile computer, provided it has a TCP/IP link with the machine running the Tivoli Workload Scheduler connector. Using the JSC, operators can schedule and administer Tivoli Workload Scheduler over the network.

- Tivoli Dynamic Workload Console

  Tivoli Dynamic Workload Console is based on Integrated Solutions Console (ISC), a Web-based console that provides the basis for a graphical interface for several different Tivoli products. Tivoli Dynamic Workload Console is the interface for performing actions related to event-driven scheduling and can also be used to generate custom reports.

In the next sections we provide an overview of the Tivoli Workload Scheduler network, workstations, and the topology used to describe the architecture in Tivoli Workload Scheduler.

2.2.1 Tivoli Workload Scheduler network

A Tivoli Workload Scheduler network is made up of the workstations, or CPUs, on which jobs and job streams are run.

A Tivoli Workload Scheduler network contains at least one Tivoli Workload Scheduler domain, the master domain, in which the master domain manager is the management hub. The master domain manager manages the databases, and from the master domain manager, you can define new objects in the databases. Additional domains can be used to divide a widely distributed network into smaller, locally managed groups.

In the simplest configuration, the master domain manager maintains direct communication with all of the workstations (fault-tolerant agents) in the Tivoli Workload Scheduler network. All workstations are in the same domain, MASTERDM (see Figure 2-1).

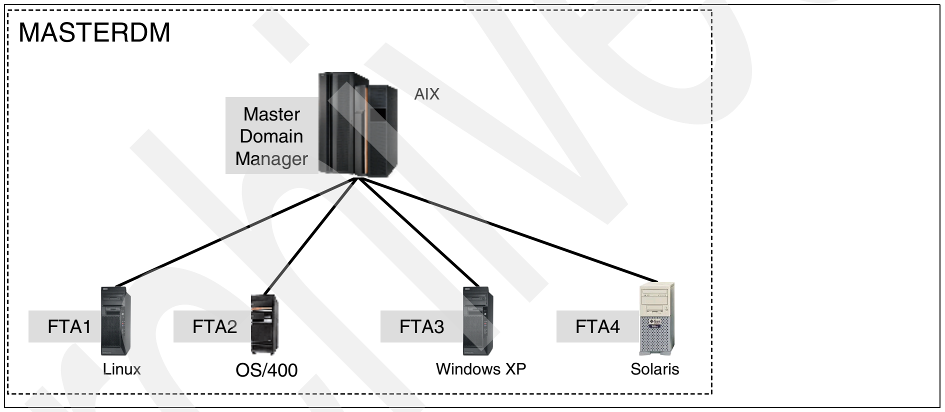


Figure 2-1 Tivoli Workload Scheduler network with only one domain

Using multiple domains reduces the amount of network traffic by reducing communications among the master domain manager (MDM) and other computers in the network. Figure 2-2 depicts an example of a Tivoli Workload Scheduler network with three domains. In this example, the master domain manager is shown as an AIX® system. The MDM does not have to be on an AIX system; it can be installed on any of several different platforms, including AIX, Linux, Solaris™, HPUX, and Microsoft Windows XP. This figure is an example that is intended to depict a typical Tivoli Workload Scheduler network.

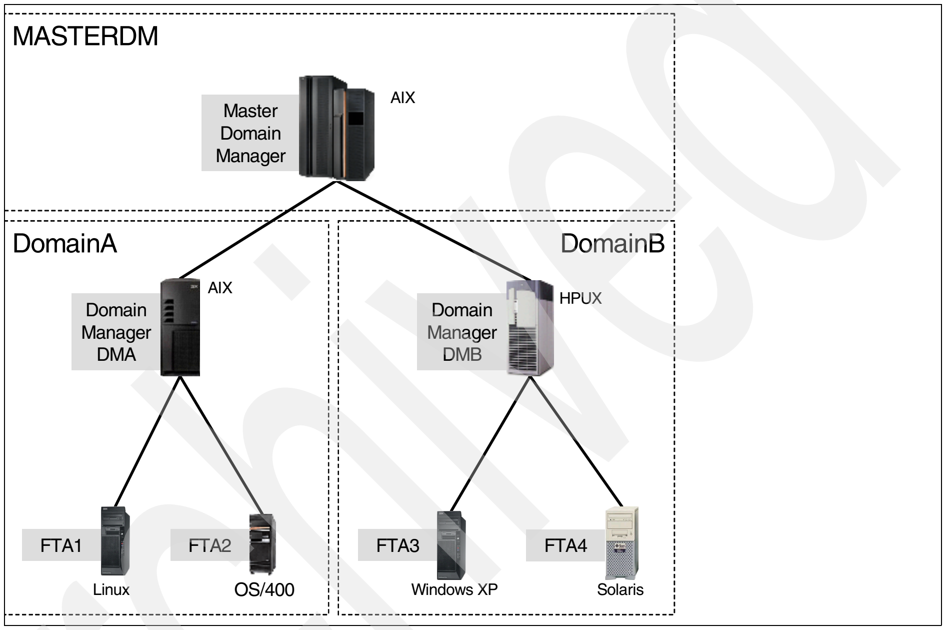


Figure 2-2 Tivoli Workload Scheduler network with three domains

In this configuration, the MDM communicates directly only with the subordinate domain managers. Each subordinate domain manager in turn communicates directly with the workstations in its domain. In this way, the number of connections from the MDM are reduced. Multiple domains also provide fault tolerance: If the link from the master is lost, a domain manager is still able to manage the workstations in its domain and resolve dependencies among them. This limits the impact of a network outage. Each domain may also have one or more backup domain managers that can become the domain manager for the domain if the domain manager fails.

Before the start of each day, the MDM creates a plan for the next 24 hours. (The plan extension is not limited to the default 24 hours. The plan covers shorter or longer periods.) This plan is placed in a production control file, named Symphony. Tivoli Workload Scheduler is then restarted throughout the network, and the MDM sends a copy of the Symphony file to each of the subordinate domain managers. Each domain manager then sends a copy of the Symphony file to the fault-tolerant agents (FTAs) in that domain.

When the network has been started, scheduling events such as job starts and completions are passed from each workstation to its domain manager. The domain manager updates its Symphony file with the events and then passes the events up the network hierarchy to the MDM. The events are then applied to the Symphony file on the MDM. Events from all workstations in the network are passed up to the MDM. In this way, the master's Symphony file contains the authoritative record of what has occurred during the production day. The master also broadcasts the changes down throughout the network, updating the Symphony files of domain managers and FTAs running in full status mode.

It is important to remember that Tivoli Workload Scheduler does not limit the number of domains or levels (that is, the hierarchy) of network. A given computing environment can have as many levels of domains as is appropriate. The number of domains or levels in the network should be based on the topology of the physical network where Tivoli Workload Scheduler is installed. Most often, geographical boundaries are used to determine divisions between domains.

Figure 2-3 shows an example of a four-tier Tivoli Workload Scheduler network, which includes the following:

- Master domain manager, MASTERDM
- DomainA and DomainB
- DomainC, DomainD, DomainE, FTA1, FTA2, and FTA3
- FTA4, FTA5, FTA6, FTA7, FTA8, and FTA9

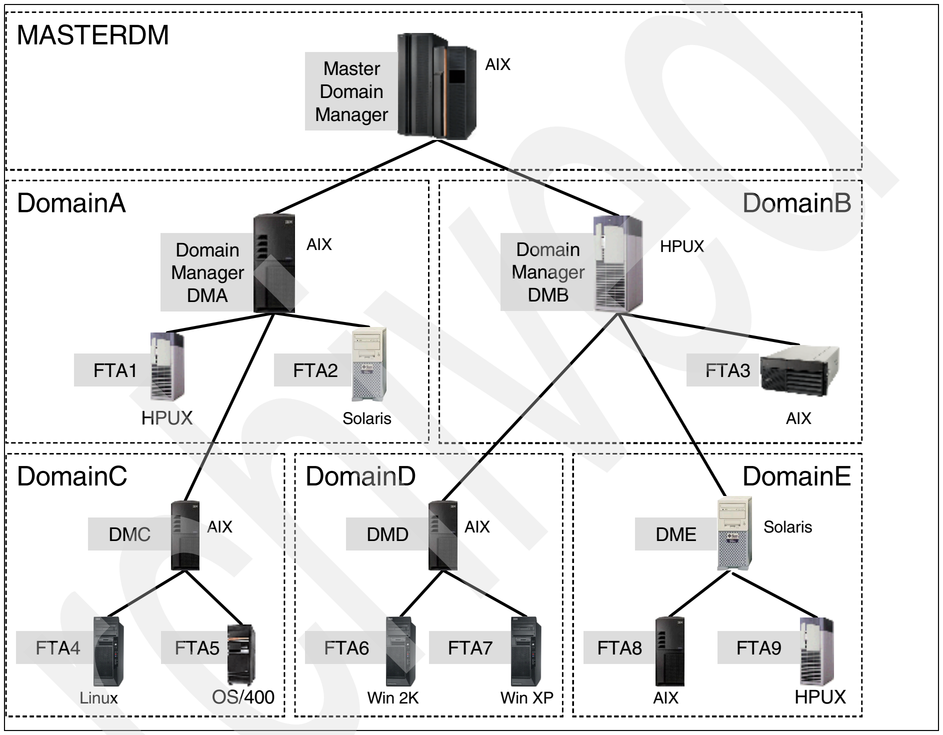


Figure 2-3 A multitiered Tivoli Workload Scheduler network

Note: Workstations are generally grouped together into a domain because they share a common set of characteristics. Most often, workstations are grouped into a domain because they are in close physical proximity to one another - for example, in the same office. Domains may also be based on organizational unit (a department, for example), business function, or application. Grouping together related workstations in a domain reduces the amount of information that must be communicated among domains, and thereby reduces the amount of network traffic.

2.2.2 Tivoli Workload Scheduler workstation types

For most cases, workstation definitions refer to physical workstations. However, in the case of extended and network agents, the workstations are logical definitions that must be hosted by a physical Tivoli Workload Scheduler workstation.

Several different types of Tivoli Workload Scheduler workstations can be used:

- Master domain manager (MDM)

  The MDM is the domain manager of the topmost domain of a Tivoli Workload Scheduler network. It contains the centralized database of all defined scheduling objects, including all jobs and their dependencies. It creates the plan at the start of each day and performs all logging and reporting for the network. The master distributes the plan to all subordinate domain managers and FTAs. In an end-to-end scheduling network, the Tivoli Workload Scheduler for z/OS engine (controller) acts as the master domain manager.

- Domain manager (DM)

  The DM serves as the management hub in a domain. All communications to and from the agents in a domain are routed through the domain manager. The DM can resolve dependencies between jobs in its subordinate agents. The copy of the plan on the DM is updated with reporting and logging from the subordinate agents.

  DMs are also used in cases where the MDM and multiple agents are separated by a firewall. Deploying a DM between the master domain manager and agents in this scenario enables you to minimize the number of open ports on the firewall.

- Backup domain manager (BDM)

  The BDM is a fault-tolerant agent capable of assuming the responsibilities of its domain manager. The copy of the plan on the BDM is updated with the same reporting and logging information that is on the DM plan.

- Fault-tolerant agent (FTA)

  A FTA is a workstation capable of resolving local dependencies and launching its jobs in the absence of a DM. It has a local copy of the plan generated in the MDM. It is also called a *fault-tolerant workstation*.

- Standard agent (SA)

  A workstation that launches jobs only under the direction of its domain manager.

- Extended agent (XA)

  An XA is a logical workstation definition that enables you to launch and control jobs on other systems and applications. Tivoli Workload Scheduler for Applications includes extended agent methods for the following systems: SAP R/3®, Oracle applications, CA7, JES2, and JES3.

Figure 2-4 shows a Tivoli Workload Scheduler network with some of the different workstation types.

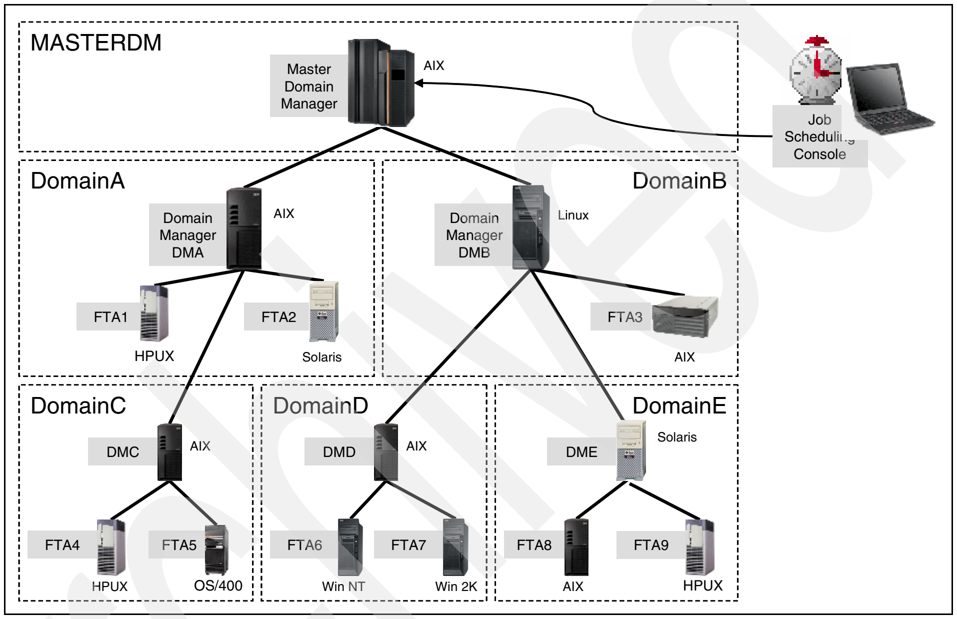


Figure 2-4 Tivoli Workload Scheduler network with workstation types

It is important to remember that domain manager FTAs, including the master domain manager FTA and backup domain manager FTAs, are FTAs with some extra responsibilities. The servers with these FTAs can be, and most often are, servers where you run normal batch jobs scheduled and tracked by Tivoli Workload Scheduler. This means that these servers do not have to be dedicated to Tivoli Workload Scheduler: they can do other work and run other applications as well.

On the other hand, you should not use one of your busiest servers as one of your Tivoli Workload Scheduler first-level domain managers.

Naming convention: When naming your workstations, make sure that you follow a naming convention. Using a naming convention has the following benefits:

- You can display information (such as frequency or department) at a glance.
- You can easily display just the workload for a single department.
- You can easily assign Security access by allowing the use of wildcards.

2.3 Supported operating systems

IBM maintains a matrix on its Web site showing the supported operating systems for all Tivoli products:

`http://www.ibm.com/software/sysmgmt/products/support/Tivoli_Supported_Platforms.html`

Click the link to check the latest supported operating system information.

2.3.1 Engine

The engine is the core Tivoli Workload Scheduler software and includes the master domain manager, backup master domain manager, and the fault-tolerant agent. In brief, the supported operating systems are:

- IBM AIX
- IBM i5/OS
- HP-UX
- Microsoft Windows
- RHEL
- Solaris
- SUSE®

Full details of the supported operating systems, patches and libraries are listed in the online document *Detailed system requirements for IBM Tivoli Workload Scheduler, version 8.4*:

`http://www-1.ibm.com/support/docview.wss?rs=672&uid=swg27010396`

This document also lists the supported processors.

2.3.2 Job Scheduling Console

As described previously, the Job Scheduling Console (JSC) is the Java-based graphical user interface for the Tivoli Workload Scheduler suite. The Job Scheduling Console runs on any machine from which you want to manage Tivoli Workload Scheduler plan and database objects.The JSC may be installed on the following operating systems:

- IBM AIX
- HP-UX
- Microsoft Windows
- RHEL
- Solaris
- SUSE.

Full details of the supported operating systems, patches, and libraries are listed in the online document *Detailed system requirements*. This document also lists the processors that are supported. You can access the document at:

http://www-1.ibm.com/support/docview.wss?rs=672&uid=swg27010396

This information is also contained in Chapter 2 of the *IBM Tivoli Workload Scheduler Job Scheduling Console User's Guide V8.4*, SC32-1257.

2.3.3 Tivoli Dynamic Workload Console

Tivoli Dynamic Workload Console is the interface for performing actions related to event-driven scheduling and can also be used to generate custom reports. Tivoli Dynamic Workload Console may be installed on the following operating systems:

- IBM AIX
- HP-UX
- Microsoft Windows
- RHEL
- Solaris
- SUSE

Full details, including the supported processors, is given in Chapter 1 of the *IBM Tivoli Dynamic Workload Console Installation and Troubleshooting Guide,* SC32-1572.

2.4 Hardware requirements

The following sections list the hardware requirements for Tivoli Workload Scheduler.

2.4.1 Engine

The online *Detailed system requirements* document lists current information about the following requirements for the Tivoli Workload Scheduler components:

http://www-1.ibm.com/support/docview.wss?rs=672&uid=swg27010396

- Master domain manager (MDM)
- Backup master domain manager (BMDM)
- Fault-tolerant agent (FTA)
- Connector for distributed engine (CONN)
- Command-line client (CLI)

This information is also available in the *IBM Tivoli Workload Scheduler Planning and Installation Guide V8.4*, SC32-1273.

Disk space requirements

Table 2-1 summarizes the minimum disk space needed for Tivoli Workload Scheduler components and features on supported operating systems.

Table 2-1 Disk space for Tivoli Workload Scheduler components

Operating system	MDM with DB2 server[a]	MDM/ BKM with DB2 client[b]	MDM/ BKM[b]	DB2 server standalone[c]	FTA	CLI	CONN
IBM AIX	1375	475	540	240	210	90	330
HP-UX	1595	655	555	240	275	90	280
Linux	1245	500	530	240	180	90	350
Solaris	1525	600	600	240	210	90	390
Solaris I386					60	9	
Windows	855	465	440	240	110	90	330

Operating system	MDM with DB2 server[a]	MDM/ BKM with DB2 client[b]	MDM/ BKM[b]	DB2 server standalone[c]	FTA	CLI	CONN
i5/OS					30		

a. These figures include the presence of the base structure for Tivoli Workload Scheduler in the DB2 database.
b. These figures do not include the presence of the base structure for Tivoli Workload Scheduler in the DB2 database because this is indicated in either the entry for the MDM with a DB2 server or the entry for the standalone DB2. These figures can be used to exclude DB2 if you are using an Oracle database.
c. The space listed in this column is divided as follows:
- 200 MB in the DB2 installation path
- 20 MB in the Data tablespace path
- 20 MB in the Reports tablespace path

Note: Where an Oracle, rather than DB2, database is to be used, the disk space requirements for DB2 should be ignored and the appropriate Oracle documentation consulted instead.

Temporary storage

Temporary file space is needed during the installation of Tivoli Workload Scheduler (see Table 2-2).

Table 2-2 Temporary file space in megabytes

Operating System Type	MDM/BKM	FTA	CLI
UNIX	170	40	40
Microsoft Windows	70	20	20

Virtual memory

In addition to permanent and temporary storage, space is needed for virtual memory (sometimes called *page file* or *swap space*). The installation requires at least 512 MB of virtual memory on any operating system.

Storage requirements during operation of Tivoli Workload Scheduler

The online *Detailed system requirements* document includes a discussion about the disk space required to operate Tivoli Workload Scheduler.

Memory requirements

Minimum and recommended memory requirements are provided in Table 2-3.

Table 2-3 Memory requirements in megabytes

Memory	MDM/BKM	FTA	CLI	CONN
Recommended	2048	256	256	1536
Required	1024	256	256	1024

2.4.2 Job Scheduling Console

Permanent storage, temporary storage, and memory requirements for the Job Scheduling Console are provided in Chapter 2 of the *IBM Tivoli Workload Scheduler Job Scheduling Console User's Guide Version 8.4*, SC32-1257, and are reproduced here.

Disk space requirements

Table 2-4 lists the permanent storage requirements in megabytes.

Table 2-4 Permanent disk space requirements

Operating system	JSC	Language pack (all languages)
IBM AIX	100	60
HP-UX	250	60
Linux on x86 and x86-64	110	60
Linux on POWER™	110	60
Solaris Operating Environment	120	60
Microsoft Windows	100	60

Table 2-5 lists the temporary disk space requirements in megabytes during the installation of the Job Scheduling Console.

Table 2-5 Temporary disk space requirements

Operating system	Temporary storage
IBM AIX	65
HP-UX	210
Linux on x86 and x86-64	75

Operating system	Temporary storage
Linux on POWER	75
Solaris Operating Environment	90
Microsoft Windows	60

Memory requirements

Minimum memory for all operating systems is 256 MB, while recommended memory is 512 MB.

2.4.3 Tivoli Dynamic Workload Console

Permanent storage, temporary storage, and memory requirements are given in Chapter 1 of the *IBM Tivoli Dynamic Workload Console Installation and Troubleshooting Guide,* SC32-1572*.*

Disk space requirements

The disk space requirements are provided in this section. The following disk space is required:

- Permanent storage: 900 MB for all operating systems.
- Temporary storage: 350 MB for all operating systems.
- Virtual memory: Swap space should be twice the physical memory.
- Storage requirements during operation of Tivoli Dynamic Workload Console: Refer to *System Requirements* for information about required disk space during use of Tivoli Dynamic Workload Console:

  `http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?topic=/com.ibm.tivoli.itws.doc/welcome.htm`

Memory requirements

The memory requirements for the Tivoli Dynamic Workload Console are 1.5 GB for all operating systems.

2.5 Software requirements

The following sections provide information about the software requirements of Tivoli Workload Scheduler.

2.5.1 Engine

Tivoli Workload Scheduler requires the use of either a DB2 or an Oracle relational database. The online *Detailed system requirements* document lists the versions of DB2 and Oracle that are currently supported.

Note: Tivoli Workload Scheduler is supplied with a version of the DB2 server and client for the supported operating systems. The operating systems that are supported by the bundled DB2, as well as the required operating system patches and libraries, are listed in the *Detailed system requirements* document.

Tivoli Workload Scheduler uses the embedded version of WebSphere Application Server Version 6.1. All operating systems supported by Tivoli Workload Scheduler are supported by the application server. It is installed automatically.

Note: In the information relating to hardware and memory requirements (see 2.4, "Hardware requirements" on page 22 and 2.5, "Software requirements" on page 26), the application server is treated as part of Tivoli Workload Scheduler, and its requirements are not listed separately.

2.5.2 Tivoli Dynamic Workload Console

The software requirements for Tivoli Dynamic Workload Console are discussed in the sections that follow.

Application server

Tivoli Dynamic Workload Console can be installed with the embedded version of WebSphere Application Server Version 6.1, that comes with the installation CD, or instead, you can use an existing WebSphere Application Server instance. See Chapter 1 of the *IBM Tivoli Dynamic Workload Console Installation and Troubleshooting Guide,* SC32-1572.

Web browsers

Mozilla Firefox and Microsoft Internet Explorer® are supported for the installation of Tivoli Dynamic Workload Console using the launchpad. Firefox and Microsoft Internet Explorer are supported as client browsers using Tivoli Dynamic Workload Console. Full details are given in Chapter 1 of the *IBM Tivoli Dynamic Workload Console Installation and Troubleshooting Guide,* SC32-1572*.*

Tivoli Workload Scheduler z/OS connector

Tivoli Workload Scheduler z/OS connector Version 8.3 fix pack 3 must be installed as an upgrade on top of Tivoli Workload Scheduler z/OS connector Version 8.3 fix pack 2. See Chapter 1 of the *IBM Tivoli Dynamic Workload Console Installation and Troubleshooting Guide,* SC32-1572 for details.

2.6 Interoperability

For information about the component levels of Tivoli Workload Scheduler that are able to work with each other, see the interoperability tables in the Release Notes available at:

`http://www-1.ibm.com/support/docview.wss?rs=672&uid=swg27010680`

3

Installation

In this chapter we provide guidelines for installing Tivoli Workload Scheduler in your environment. The following topics are discussed in this chapter:

- “Installation planning” on page 30
- “Installing a Tivoli Workload Scheduler V8.4 instance” on page 32
- “Upgrading masters from a previous version to V8.4” on page 36
- “Upgrading agents” on page 40

3.1 Installation planning

Before installing Tivoli Workload Scheduler attend to several considerations. These considerations are discussed in the sections that follow.

3.1.1 Embedded version of IBM WebSphere Application Server

Connection between the Tivoli Workload Scheduler engine, the Job Scheduling Console, and the command line requires the embedded version of IBM WebSphere Application Server V6.1. The embedded version of IBM WebSphere Application Server V6.1 is installed automatically when you install a master domain manager (MDM) or backup master domain manager (BMDM). In this chapter, the embedded version of IBM WebSphere Application Server V6.1 is referred to as WebSphere Application Server.

3.1.2 Supported Relational Database Management Systems

A Relational Database Management System (RDBMS) is required on master domain managers and their backups. The RDBMS can be one of the following:

- DB2 Enterprise Server Edition
- Oracle

DB2 considerations

The supported DB2 Enterprise Server edition versions are:

- DB2 Server V8.2 (DB2 Server V8.1 with fix pack 7) For enhanced performance, use DB2 8.2.4 (V8.1 fix pack 1).
- On Solaris 10 environments: DB2 Server V8.2.2 (DB2 Server V8.1 with fix pack 9).
- DB2 Server V9.1.

DB2 Server V9.1 is bundled with this version of the product (located on Tivoli Workload Scheduler Installation disk 4). Note that DB2 Enterprise Server does not have to be installed on the same computer as the master domain manager (MDM). In fact, to provide the most resilience, you should install DB2 on a separate database server. In this case you must install the DB2 administration client on the MDM to access the computer where the DB2 server is installed.

Oracle considerations

Oracle Database is the other choice of required RDBMS for installing an MDM. The Oracle Database versions both for server and client required by Tivoli Workload Scheduler can be one of the following:

- Oracle Database 9i Release 2 - Enterprise Edition (V9.2.0.x) or later
- Oracle Database 10g Release 2 - Enterprise Edition (V10.2.0.x) or later

The Oracle Database must have the partitioning option installed.

The database must be already installed and configured before you install Tivoli Workload Scheduler. Before you start installation of an MDM, make sure you have the following information:

- The path pointing to where the Oracle installation is located.
- The path must identify a tree in the Oracle structure that includes the sqlplus executable.
- The net service name of the database. It must have been already created by the Oracle administrator.
- The name and password of the Oracle administrator.
- The name of the Tivoli Workload Scheduler data and temporary tablespaces. They must have been already created by the Oracle administrator.

Important: If an existing user is specified as the Tivoli Workload Scheduler Oracle user when installing a fresh instance of a Tivoli Workload Scheduler V8.4 master domain manager, the installation process assumes that the schema is at the right level and does not create the database objects (tables, views, clusters, procedures, indexes, and so on).

3.1.3 Tivoli Workload Scheduler user considerations

The following sections provide details on Tivoli Workload Scheduler user considerations.

Microsoft Windows

Windows users for standalone workstations (not part of a domain) require the following permissions:

- Is a member of the administrative group
- Has the Act as part of the operating system privilege

- Has the Log on as a service privilege, if the server is run as a service
- Has the Impersonate a client after authentication right

Microsoft Windows domain users require the following permissions:

- Act as part of the operating system
- Log on:
  - Locally
  - As batch job
  - As service
- Replace process level token
- Impersonate a client after authentication right

UNIX and Linux

On UNIX and Linux operating systems, regardless of the method of installation you choose, the Tivoli Workload Scheduler user must be created manually before running the installation. Use the appropriate UNIX and Linux operating system commands to create the user.

3.1.4 Multiple instances

Multiple copies of the product can be installed on a single computer provided that a unique name and installation path is used for each instance. Instances are recorded in the registry file.

3.2 Installing a Tivoli Workload Scheduler V8.4 instance

You can use several installation methods to install Tivoli Workload Scheduler. These methods are discussed in the sections that follow.

3.2.1 InstallShield wizard

Tivoli Workload Scheduler master domain managers, backup masters, agents, and connectors can be installed by launching the individual setup files for each supported platform.

You can use the installation wizard in interactive or silent mode, which is called the ISMP (InstallShield MultiPlatform) method. In interactive mode, the wizard guides you through the installation steps. In silent mode, a response file provides the relevant information to the installation process, which is run in the background. The ISMP silent installation method is especially important when deploying a large number of fault-tolerant agents (FTAs).

Note: If you are using IBM Tivoli Configuration Manager or IBM Tivoli Provisioning Manager, you can also use the software package block method, described in 3.2.4, "Software distribution software package blocks" on page 36, to deploy large numbers of FTAs to remote locations.

To install a new instance of Tivoli Workload Scheduler, perform the following steps:

1. Insert the installation CD according to the operating system you are running.
2. Run setup for the operating system on which you are installing:
   - On Microsoft Windows: WINDOWS\SETUP.exe
   - On UNIX and Linux: SETUP.sh or operating_system/SETUP.bin, depending on whether you want to copy the images locally

Alternatively, start the launchpad as described in 3.2.2, "Launchpad" on page 34 and select the Tivoli Workload Scheduler installation.

3. After you select the installation language and accepting the license agreement, the installation wizard displays the Installation choice window. Select **Install an instance of Tivoli Workload Scheduler** to install a new instance.

Note: The Upgrade Instance and Add new features are available only if you have an existing installation of Tivoli Workload Scheduler.

4. Follow the prompts to complete the installation.

Note: Refer to*Deployment Guide Series: IBM Tivoli Workload Scheduler V8.4 and IBM Tivoli Dynamic Workload Broker V1.2*, SG24-7528, for step-by-step installation instructions on installing Tivoli Workload Scheduler components.

3.2.2 Launchpad

You can use the launchpad to guide you through the installation of all the Tivoli Workload Scheduler components from a single interface.

From the launchpad you can:

- Install or upgrade all Tivoli Workload Scheduler components
- Install the Job Scheduling Console
- Install the Tivoli Dynamic Workload Console
- Access information about Tivoli Workload Scheduler

The launchpad automatically accesses and runs the related installation setup file in interactive mode.

To access the launchpad, one of the following Web browsers is required:

- Mozilla, V1.7 and later
- Firefox, V1.0 and later
- Microsoft Internet Explorer (only for Microsoft Windows operating systems) Version 5.5 and later

To start the launchpad installation program, perform the following steps:

1. Insert the CD containing the required installation image into the CD drive.
2. The procedure for starting the launchpad varies depending on the operating system:

- On Microsoft Windows systems: From the root directory of the CD, run the launchpad.exe file.
- On UNIX systems: From the root directory of the CD, run the launchpad.sh& file.

3.2.3 twsinst script for UNIX and Linux operating systems

Tivoli Workload Scheduler agents can be installed on UNIX and Linux operating systems using the twsinst script. This method does not use a Java Virtual Machine, and you can use it instead of the installation wizard. You must have root access to run the twsinst script.

The command syntax for a new installation is as follows:

```
twsinst -new -uname <username>
    [-thiscpu <cpuname>]
    [-master <cpuname>]
```

```
[-port <port_number>]
[-company <company_name>]
[-inst_dir <install_dir>]
[-lang <lang_id>]
[-create_link]
[-skip_usercheck]
[-reset_perm]
```

where:

- `-new`

  Specifies the type of installation to perform. -new means a fresh installation of the Tivoli Workload Scheduler agent or master is installed.

- `-uname <username>`

  The name of the user for which Tivoli Workload Scheduler is installed, updated, or uninstalled. This user name is not to be confused with the user performing the installation logged on as root. For a new installation, this user account must be created manually before running the installation. Create a user with a home directory. Tivoli Workload Scheduler is installed by default under the HOME directory of the specified user.

- `-thiscpu <cpuname>`

  The name of the Tivoli Workload Scheduler workstation of this installation. The name cannot exceed 16 characters. This name is registered in the localopts file. If not specified, the default value is the host name of the workstation.

- `master <cpuname>`

  The workstation name of the master domain manager. This name cannot exceed 16 characters and cannot contain spaces. If not specified, the default value is MASTER.

- `-port <port_number>`

  The TCP/IP port number. This number is registered in the localopts file. The default value is 31111.

- `-company <company_name>`

  The name of the company. The company name cannot contain blank characters. The name appears in program headers and reports. If not specified, the default name is COMPANY.

- `-inst_dir <install_dir>`

  The directory of the Tivoli Workload Scheduler installation. This path cannot contain blanks. If not specified, the path is set to the user name home directory.

- `-lang <lang_id>`

  The language in which the twsinst messages are displayed. If not specified, the system LANG is used. If the related catalog is missing, the default C language catalog is used.

- `-create_link`

  Create the symlink.

- `-skip_usercheck`

  Skip the check of the user in the /etc/password file or by using the **su** command. Enable this option if the authentication process within your organization is not standard, thereby disabling the default authentication option.

- `-reset_perm`

  Reset the permissions of the libatrc library.

Note: The twsinst script is supported only for installation or upgrade of fault-tolerant agents, not master domain managers.

3.2.4 Software distribution software package blocks

Tivoli Workload Scheduler agents can be installed using IBM Tivoli Configuration Manager or IBM Tivoli Provisioning Manager by distributing software package blocks (SPBs). In a large-scale environment, it is useful to provision the agents using IBM Tivoli Configuration Manager or IBM Tivoli Provisioning Manager. Doing so is a quick way to set up your Tivoli Workload Scheduler environment when you do not have local access to the target machines.

Important consideration about uninstallation: When uninstalling Tivoli Workload Scheduler, use the same method that was used to install the product whenever possible. For example, if you installed the product using the installation wizard, use the uninstaller program to subsequently remove the product. The uninstaller program is created during the installation procedure.

3.3 Upgrading masters from a previous version to V8.4

You can upgrade master domain managers (MDMs) and backup master domain managers (BMDMs) from Tivoli Workload Scheduler V8.2, V8.2.1, or V8.3 to Version 8.4. If you have components of Tivoli Workload Scheduler that are at an earlier version than V8.2 you must first upgrade them to V8.2 or V8.2.1 before upgrading them to Version 8.4.

Refer to the upgrade process described in *IBM Tivoli Workload Scheduler Planning and Installation Guide V8.4*, SC32-1273, for the overall upgrade methodology.

In this section, we focus on one key step: how to upgrade the database from V8.2.x to V8.4. The upgrade of the MDM from Version 8.2.x not only upgrades the component, but also imports the Version 8.2.x Mozart database into a new RDBMS, using DB2 or Oracle support. It also imports the configuration files such as the run number file and globalopts file. This process is important because in V8.2.x, these files were not kept in the RDBMS, which is the case in V8.3 and V8.4.

The main advantage of the new Version 8.4 infrastructure is that using an RDBMS avoids data inconsistencies. Furthermore, you can separate the database onto a different computer, helping to spread the workload and improving the performance of the MDM. Using an RDBMS also enables you to designate a backup master domain manager that is configured to access the same database. Switching the MDM is now a much quicker process with reduced risk of data loss. Importing the database can be performed automatically by the upgrade process or manually after the MDM is upgraded. The manual process is performed object type by object type, giving you maximum control.

3.3.1 Importing data

The data import requires that you perform the following steps. You must perform these steps irrespective of whether you are importing the data manually, or the upgrade wizard is importing it for you. If you are using the update wizard for a direct upgrade, these steps are performed automatically.

1. Import the configuration files.

   The V8.2.x instance has two configuration files that must be migrated:

   – Run number file

     This field contains the run number used by Symphony and is automatically incremented every time you run Jnextday. It is used in much the same way in Version 8.4, but to preserve continuity you must import it. Version 8.4 is supplied with the **optman miggrunnb** subcommand that imports this file.

```
optman miggrunnb TWS_8.2.x_main_dir
```

where `TWS_8.2.x_main_dir` is the root directory of the previous Tivoli Workload Scheduler version.

- Globalopts file

  This file contains the global options used by the MDM. In Version 8.4 it is replaced by a database table, but the values must be imported. Version 8.4 is supplied with the **optman miggopts** subcommand that imports this file.

  The syntax of the command is:

  ```
  optman miggopts TWS_8.2.x_main_dir
  ```

  where `TWS_8.2.x_main_dir` is the root directory of the previous Tivoli Workload Scheduler version.

2. Export the object data (except for Microsoft Windows user data) from the Mozart database.

   The object data is extracted from the V8.2.x database using a specially enhanced version of the V8.2.x composer, called composer821, which is bundled with Version 8.4. Do not use the original **composer** command for the migration. The **create** subcommand is used to create text files that contain the object data definitions.

3. Export the Microsoft Windows user data from the Mozart database.

   If you use composer or composer821 to export the Windows users in the database, they are exported without passwords. This is the normal, secure behavior of composer. For this reason, a migration utility, migrutility get_users, has been provided that extracts the Windows user data with the users' encrypted passwords. This is important, for example, if you forget the passwords used for Windows user scheduling definition objects in your Version 8.2 database.

   The syntax for the command is

   ```
   migrutility get_users TWS_8.2.x_user_mozart_file users_filename
   ```

   where:

   - `TWS_8.2.x_user_mozart_file`

     The complete path to file userdata located in *TWShome*/network/userdata

   - `users_filename`

     A name of your choice for the output

4. Export the Tivoli Management Framework user data from the security file.

   In your Version 8.2.x security file, you might have Tivoli Management Framework administrators defined (used to assign rights to users who access the Job Scheduling Console). These must be migrated by a tool that substitutes each Tivoli Management Framework administrator with a local user account associated with those users. The users are exported first using the **dumpsec** command on the V8.2.x system. Second, the migrfwkuser utility does the account substitution. The output of the migrfwkuser utility is a security file containing the migrated users and your existing settings. You must then merge this information with the new Version 8.4 security settings to build your final security file.

   The syntax of the migrfwkuser script is as follows:

   ```
   migrfwkuser -in input_security_file -out output_security_file [-cpu 
   workstation] [-hostname local_hostname]
   ```

   where:

   – `input_security_file`

     The file created using the **dumpsec** command

   – `output_security_file`

     The security file that is created by the migrfwkuser script

   – `workstation`

     The name of the local workstation where the log on data added by the tool is defined

   – `local_hostname`

     The fully qualified host name of the Tivoli Management Framework users to be extracted

5. Import the object data into the new database.

   The object and Microsoft Windows user data in the text files is imported into the new database using the datamigrate utility. This follows the stricter validation of the new database and identifies any logical discrepancies that were caused by the less strict validation in the V8.2.x system. In nearly all cases the utility is able to modify some aspect of the object and add it to the new database. A report of the discrepancies is produced so that you can fix the problems. You can choose to accept the way that the tool has dealt with the problem, or you can use the facilities of Version 8.4 to modify the object.

   For example, when mapping job definitions, if the workstation of the job does not exist, the job is not created in the database. However, if the recovery job does not exist, the job is created but a warning is issued.

The datamigrate utility includes the following options:

– Migrate directly from the Version 8.2.x database or from the text files created by composer821 and migrutility get_users.

– Migrate each object type one by one, or import all object data in one command.

6. Import the Tivoli Management Framework user data to the security file.

   The **makesec** command is used to create the final security file from the file created by the migrfwkuser utility merged with the new Version 8.4 security settings.

   Refer to *IBM Tivoli Workload Scheduler Planning and Installation Guide V8.4*, SC32-1273 for more information about these steps.

3.4 Upgrading agents

Before you perform an upgrade on an agent workstation, ensure that all Tivoli Workload Scheduler processes and services are stopped. If you have jobs that are currently running, you must manually stop the related processes.

You can upgrade Tivoli Workload Scheduler agents using several different methods, which are discussed in the sections that follow.

Using the installation wizard

In this method you start the installation program similar to the way you start a new installation. After selecting the installation language and accepting the license agreement, you select an existing installation of a previous release of the product from the drop-down list and perform the upgrade.

Using a silent installation

Alternatively you can use the silent installation. To upgrade an agent using a silent installation, you must use the following response file templates provided on the CDs in the \RESPONSEFILES\ directory:

- TWS84_UPGRADE_Agent.txt for a Tivoli Workload Scheduler V8.2.x or V8.3 agent
- TWS84_UPGRADE_Connector.txt for a Tivoli Workload Scheduler V8.3 agent with connector

You can customize the response file based on your environment:

- For a Windows agent upgrade, use the following syntax:

  ```
  SETUP.exe -options <local_dir>\response_file.txt -silent
  ```

- For a UNIX or Linux agent upgrade, use the following syntax:

  ```
  ./SETUP.bin -options <local_dir>/response_file.txt -silent
  ```

Using the twsinst script

If you use the twsinst method, the command syntax for an upgrade is the following:

```
twsinst -update -uname username
   [-inst_dir install_dir]]
   [-backup_dir backup_dir]
   [-nobackup_dir]
   [-lang <lang_id>]
   [-create_link]
   [-skip_usercheck]
   [-reset_perm]
```

where:

- `-update`

  Upgrades an existing agent installation.

- `-uname <username>`

  The name of the user for which Tivoli Workload Scheduler is being updated.The software is updated in this user's home directory. This user name is not to be confused with the user performing the upgrade.

- `inst_dir`

  The directory of the Tivoli Workload Scheduler installation. This path cannot contain blanks. If not specified, the path is set to the user name home directory.

- `-backup_dir`

  An alternative directory (which must be created manually) as the destination for the backup copy of a previous version.

- `-nobackup_dir`

  No backup is made.

- `-lang <lang_id>`

  The language in which the twsinst messages are displayed. If not specified, the system LANG is used. If the related catalog is missing, the default C language catalog is used.

- `-create_link`

  Creates the symlink.

- `-skip_usercheck`

  Skips the check of the user in the /etc/password file or by using the **su** command. Enable this option if the authentication process within your organization is not standard, thereby disabling the default authentication option.

- `-reset_perm`

  Resets the permissions of the libatrc library.

Using software distribution software package blocks

You can also use IBM Tivoli Configuration Manager or IBM Tivoli Provisioning Manager to upgrade your agents. The software package block uses a number of Tivoli Workload Scheduler parameters to perform the upgrade. You can assign values to each variable to reflect the installation that is being upgraded; otherwise, the default value is assigned.

3.4.1 Job Scheduling Console V8.4 installation

You can use the following methods to install the Job Scheduling Console:

- ISMP installation wizard

  When you are installing the Job Scheduling Console on a single workstation, you can use the installation wizard in interactive or silent mode. In interactive mode, the wizard guides you through the installation steps. Installation uses a temporary directory for ISMP and a temporary directory for the Job Scheduling Console. You can set the installation wizard temporary directory using -is:tempdir directory_name.

  In silent mode a response file provides the relevant information to the installation process, which is run in the background without user intervention. For example to install the Job Scheduling Console on a Microsoft Windows system using the silent method, you can use the following command, specifying the fully qualified path to the response file:

  ```
  setup.exe -silent -options filename
  ```

- Software distribution

  When you are installing the Job Scheduling Console on more than one workstation simultaneously, you can use a software package block (SPB) method

Backward compatibility

The earliest version of the Job Scheduling Console that can be used with Tivoli Workload Scheduler V8.4 is Version 8.3 because prior versions of the Job Scheduling Console required Tivoli Management Framework. However, if you want to use new functions that are available with Tivoli Workload Scheduler V8.4, you must use Job Scheduling Console V8.4. On the other hand, you can use Job Scheduling Console V8.4 with Tivoli Workload Scheduler V8.3.

For example, if you have an existing Tivoli Workload Scheduler V8.2 environment and plan to run a Tivoli Workload Scheduler V8.4 master in parallel until all jobs and schedules can be migrated into the new environment, two Job Scheduling Console installations are required: Job Scheduling Console V1.3 or V1.4 for the Tivoli Workload Scheduler V8.2 environments, and Job Scheduling Console V8.4 for the newly installed Tivoli Workload Scheduler V8.4 environment.

3.4.2 Tivoli Dynamic Workload Console V8.4 installation

The Tivoli Dynamic Workload Console is a Web-based user interface for the following set of products:

- Tivoli Workload Scheduler
- Tivoli Workload Scheduler for z/OS
- Tivoli Workload Scheduler for Applications
- Tivoli Dynamic Workload Broker

You can access your Tivoli Workload Scheduler environments from any location in your network using a supported browser connected to the Tivoli Dynamic Workload Console. The Tivoli Dynamic Workload Console must reside on a system that can reach the Tivoli Workload Scheduler node using network connections.

You can install the Tivoli Dynamic Workload Console on:

- The following existing WebSphere Application Server instances:
  - WebSphere Application Server Version 6.1.0.9
  - WebSphere Application Server Network Deployment Version 6.1.0.9 in the standalone application server profile mode
- Its embedded version of WebSphere Application Server Version 6.1.0.9. The embedded version of WebSphere Application Server Version 6.1.0.9, is available in the Web console installation CD, and it is installed during the Web console installation.

Tivoli Workload Scheduler supported engines

The Tivoli Dynamic Workload Console supports any of the following Tivoli Workload Scheduler distributed configurations:

- Tivoli Workload Scheduler, Version 8.4

  All new Tivoli Workload Scheduler functions available on the Web console are accessible.

- Tivoli Workload Scheduler, Version 8.3 fix pack 3

  Tivoli Workload Scheduler, Version 8.3 fix pack 3 functions are available on the Tivoli Dynamic Workload Console are accessible except for the reporting and event management capabilities.

4

Configuration

This chapter describes different configuration actions that tailor Tivoli Workload Scheduler to a specific environment and to local requirements after Tivoli Workload Scheduler has been installed.

The following configuration topics are described in this chapter:

4.1 Configuring a Tivoli Workload Scheduler instance

When the Tivoli Workload Scheduler installation procedure has been completed on a given server, it is necessary to configure the installed Tivoli Workload Scheduler instance as a Tivoli Workload Scheduler workstation in the Tivoli Workload Scheduler network.

The installed Tivoli Workload Scheduler workstation can, for example, be configured as a master domain manager (MDM), a backup master domain manager (BMDM), or a fault-tolerant agent (FTA). The type of workstation defines which role the workstation performs in the Tivoli Workload Scheduler network (see Chapter 2, “Planning” on page 11).

During configuration of the Tivoli Workload Scheduler workstation, make sure that the workstation is started and stopped when the server is started and stopped. In addition, make sure that the Tivoli Workload Scheduler workstation can communicate with other workstations in the Tivoli Workload Scheduler network.

This chapter describes configuration tasks you can perform to configure a Tivoli Workload Scheduler workstation.

4.1.1 Configuring a master domain manager workstation

After you install the Tivoli Workload Scheduler master domain manager (MDM), you must configure it to be able to produce (or extend) a new production plan on a daily basis.

The production plan (or just *plan)* is handled and extended automatically by jobs in a job stream named *FINAL.* When the FINAL job stream has been added to the database and **JnextPlan** has been run once, the FINAL job stream is placed in the plan every day and runs the jobs required to generate a new plan (or extend the current plan).

The installation creates a file Sfinal in the TWShome directory on the server where the Tivoli Workload Scheduler MDM has been installed.

If the option to automatically add the FINAL job stream during installation was not selected, the FINAL job stream must be added manually to the Tivoli Workload Scheduler database. The steps in this section describe how to add the FINAL job stream to the database and run the **JnextPlan** command manually for the first time.

The following is an example of how to configure a master domain manager after installation:

1. Log on as TWSuser.
2. Set the environment variables, using the tws_env.sh script on UNIX (or tws_env.cmd on Microsoft Windows), in the TWShome directory.
3. Add the FINAL job stream to the Tivoli Workload Scheduler database by running the **composer** command:

   ```
   composer add Sfinal
   ```
4. Run the JnextPlan job: `JnextPlan`.
5. When JnextPlan completes, check the status of the Tivoli Workload Scheduler:

   ```
   conman status
   ```

   If the Tivoli Workload Scheduler is started correctly, the status returned by the command is as follows:

   ```
   Batchman LIVES
   ```
6. Raise the workstation limit value to allow jobs to run. The default job limit after installation is 0, so no jobs can run. Raise the job limit to enable jobs to run; for example, to run 10 jobs concurrently:

   ```
   conman "limit ;10"
   ```

   If no workstation name is specified for the **conman limit** command, the default value is the current logon workstation.

Note: The FINAL job stream has changed in Tivoli Workload Scheduler Version 8.4. Remember to replace the “old” existing FINAL job stream in the database with the Tivoli Workload Scheduler Version 8.4 FINAL job stream if you are upgrading from previous versions of Tivoli Workload Scheduler.

The Tivoli Workload Scheduler Version 8.4 FINAL job stream and jobs in the job stream are shown in Figure 4-1.

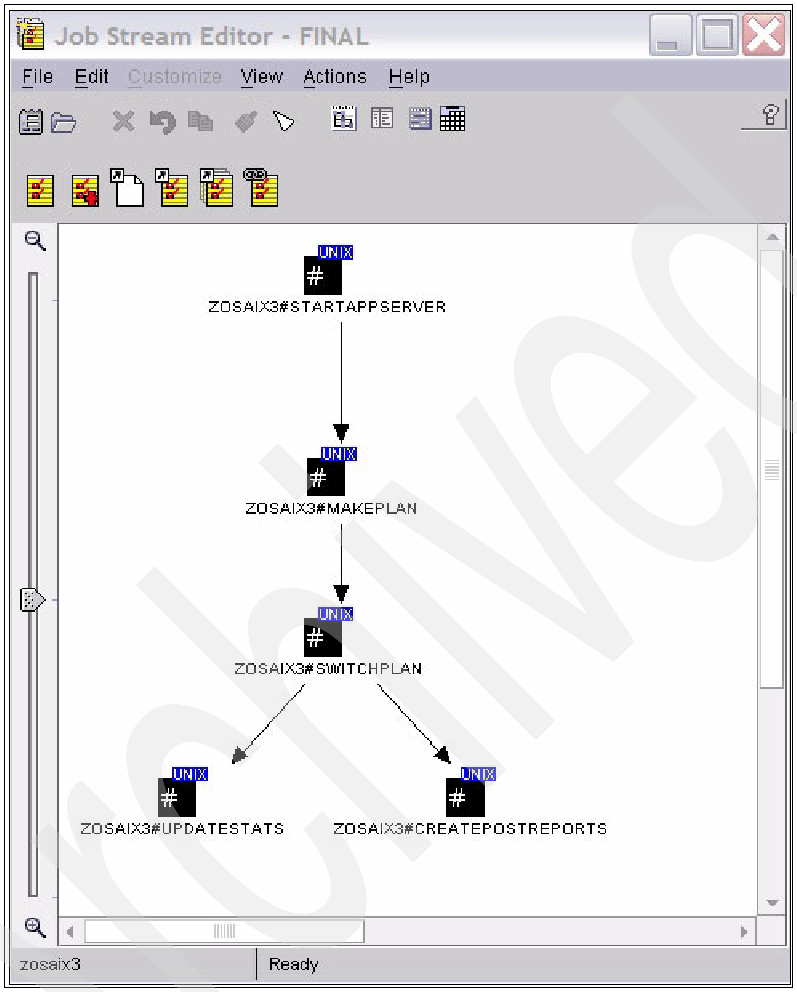


Figure 4-1 Tivoli Workload Scheduler Version 8.4 FINAL job stream and jobs

For more details about how the FINAL job stream jobs have changed in Tivoli Workload Scheduler Version 8.4, see 5.2.2, "Creating the production plan" on page 114.

4.1.2 Configuring a backup master domain manager workstation

Perform the following configuration steps after the Tivoli Workload Scheduler backup master domain manager has been installed:

1. Log on as TWSuser on your master domain manager.
2. Add the backup master user name and password to the useropts file (see 4.6.3, "Defining options for using the user interfaces" on page 97).
3. Set the environment variables, using the tws_env.sh script on UNIX (or tws_env.cmd on Microsoft Windows) in the TWShome directory.
4. Define the DBM workstation in the Tivoli Workload Scheduler database on the MDM using the composer command interface or the Job Scheduling Console*.*

   The following example demonstrates using the composer command-line interface:

   a. Enter **`composer new`** to open a text editor.

   b. Type the workstation definition, for example:

   ```
   CPUNAME BMDM
     DESCRIPTION "Backup Master Domain Manager workstation"
     OS UNIX
     NODE domain1 TCPADDR 31111
     DOMAIN MASTERDM
     FOR MAESTRO
       TYPE FTA
       AUTOLINK ON
       BEHINDFIREWALL OFF
       FULLSTATUS ON
   END
   ```

   Note: TCPADDR is the TCP/IP port number Tivoli Workload Scheduler uses for communication between workstations. NODE is the server host name or IP address.

   Note: It is important to specify FULLSTATUS ON for the backup master domain manager.

5. Wait for the FINAL job stream to run or run the following:

   ```
   JnextPlan -for 0000
   ```

   to include and activate the backup master workstation in the plan and send a Symphony file to the backup master workstation server.

6. Change the workstation limit to enable jobs to run on the workstation. For example, set the number of jobs to run concurrently on the workstation to 10 by entering the following:

   ```
   conman “limit BMDM;10”
   ```

4.1.3 Configuring an agent or domain manager workstation

After installing an agent or a domain manager on a server, you must define the corresponding workstation and domain (if you installed a domain manager) in the Tivoli Workload Scheduler database. In addition, you must verify that the workstation can be linked from the master domain manager workstation.

You can configure agents or domain managers using the composer command-line interface or from the Job Scheduling Console.

Configuring a domain manager workstation

This section provides an example that demonstrates how to configure a domain manager using the command-line interface. Do so by performing the following steps:

1. Log on to the MDM as TWSuser.
2. Set the environment variables, using the tws_env.sh script on UNIX (or tws_env.cmd on Microsoft Windows) in the TWShome directory.
3. Create the domain and workstation definitions in the Tivoli Workload Scheduler database:
   a. Enter **`composer new`** to open a text editor.
   b. Type the domain and workstation definitions, for example:

   ```
   DOMAIN DOMAIN3
    DESCRIPTION "DOMAIN 3"
    PARENT MASTERDM
   END

   CPUNAME DM3
     DESCRIPTION "Domain Manager for Domain 3"
     OS UNIX
     NODE domain1 TCPADDR 31111
     DOMAIN DOMAIN3
   ```

```
  FOR MAESTRO
    TYPE MANAGER
    AUTOLINK ON
    BEHINDFIREWALL OFF
    FULLSTATUS ON
END
```

Note: In the preceding script, you can determine from `TYPE` that the workstation is a manager (domain manager) and the workstation is manager for domain `DOMAIN3`. `DOMAIN3` must be defined before the workstation (`CPUNAME`), which is done by the DOMAIN definition.

Note: With Tivoli Workload Scheduler Version 8.3 and later versions, the name of the master domain is not restricted to MASTERDM. This means that it is not possible to decide if a workstation displayed using composer or the Job Scheduling Console is the master domain manager or a domain manager; for both types, the workstation type is MANAGER.

To determine whether the workstation is a manager or master domain manager, you must display or list the domain definition using composer or the Job Scheduling Console.

Configuring an agent workstation

You can configure three types of agent workstations in the Tivoli Workload Scheduler:

- Fault-tolerant agent workstation (type = fta)

  A workstation capable of resolving local dependencies and launching its jobs in the absence of a domain manager. It has a local copy of the plan generated in the master domain manager.

- Standard agent workstation (type = s-agent)

  A workstation that launches jobs only under the direction of its domain manager. Standard agents are not fault tolerant.

- Extended agent (type = x-agent)

  The extended agent workstation type is described in 4.5, “Configuring extended agents” on page 70.

The following is an example procedure that demonstrates how you can configure a Tivoli Workload Scheduler fault-tolerant agent (FTA):

1. Log on to the MDM as TWSuser.
2. Set the environment variables, using the tws_env.sh script on UNIX (or tws_env.cmd on Microsoft Windows) in the TWShome directory.
3. Create the fault-tolerant workstation definition in the Tivoli Workload Scheduler database by completing these steps:
   a. Enter **`composer new`** to open a text editor.
   b. Enter the domain and workstation definitions, for example:

```
CPUNAME FTA1
  DESCRIPTION "Fault Tolerant Workstation"
  OS UNIX
  NODE ftadnsname TCPADDR 31111
  DOMAIN MASTERDM
  FOR MAESTRO
    TYPE FTA
    AUTOLINK ON
    BEHINDFIREWALL OFF
    FULLSTATUS OFF
END
```

When configuring FTAs, standard agents, and domain managers, you must remember that the workstations (that is, the agents) must be able to establish a two-way network link to and from the MDM (or domain manager if used).

For example, to establish a two-way link between a FTA and its master domain manager, the following requirements must be satisfied:

- The master domain manager can resolve the FTA node information:
  – The NODE option in the workstation definition contains the server host name or IP address for the FTA.
  – The NODE information is used by the MDM to establish a downward link to the FTA.

- The FTA can resolve the MDM node information.

  When the FTA receives the Symphony file, FTA tries to contact the master domain manager workstation and tries to establish an upward link to the MDM using the server host name or IP address specified in the NODE keyword for the MDM workstation.

- The netman process is running on the FTA and on the MDM and is using the port specified in the TCPADDR option for the current production day.

  NODE and TCPADDR specify the host name and port number that the FTA and MDM use to establish the two-way network link.

> **Note:** To check or verify which netman port (specified in the TCPADDR option) and node is being used for the current production day, use the following command:
>
> ```
> conman "sc;l"
> ```
>
> where
>
> ```
> sc=showcpu
> l=link
> ```
>
> Doing so, TWS displays all workstations in the Symphony file (plan) and their link information.

Fault-tolerant agent or standard agent

The standard agent’s function is similar to the function of the FTA, yet the former lacks both local fault tolerance and local job stream and job launch capability. Standard agents rely on their host FTAs to resolve dependencies and to manage a local job stream and job launch.

The lack of fault tolerance has caused many organizations to select FTAs over standard agents, but standard agents do have merits in certain cases, for example:

- When system resources are not sufficient to host an FTA.

  Standard agents do not run Batchman, so they require less resources than FTAs.

- The Symphony file sent to a standard agent contains only workstation and calendar information.

  This limited information considerably shrinks the Symphony file sent to standard agents and reduces the time required for Symphony distribution.

- In cluster environments with many agents, it is often easier to use standard agents.

Note: If you decide to use standard agents, make sure you have a high-speed link between the standard agents and their domain managers.

4.1.4 Starting Tivoli Workload Scheduler on UNIX and Linux

Another important configuration task for Tivoli Workload Scheduler workstations installed on UNIX or Linux operating systems is to make sure that the Tivoli Workload Scheduler workstation is automatically started when the server boots up. The Tivoli Workload Scheduler installation program does not perform actions that accomplish this task.

You can automatically start the Tivoli Workload Scheduler in UNIX and Linux by invoking the Tivoli Workload Scheduler StartUp command from the /etc/inittab file as shown in Example 4-1.

Example 4-1 Simple StartUp script to automatically start Tivoli Workload Scheduler

```
if [-x TWShome/StartUp]
then
echo "netman started..."
/bin/su - TWSuser -c " TWShome/StartUp"
fi
```

Important: The StartUp script must be run by the TWSuser!

The StartUp script starts the Tivoli Workload Scheduler netman process, and if the script is issued on a master domain manager where WebSphere Application Server is installed, the StartUp script also starts the WebSphere Application Server for Tivoli Workload Scheduler.

The remainder Tivoli Workload Scheduler process tree can be started with the following commands:

```
conman start
conman startmon
```

You must remember that the StartUp script starts only after the netman process and eventually WebSphere Application Server process completes.

Note: On Microsoft Windows workstations, the StartUp command (that is, the script) starts Tivoli Netman and Tivoli Token Service. It also starts WebSphere Application Server if installed together with Tivoli Workload Scheduler. These services are started automatically when the Windows server is booted.

4.2 Controlling job-scheduling environment with jobmanrc

Sometimes it is necessary to control or manage the scheduling environment for jobs to be executed on a specific Tivoli Workload Scheduler workstation, for example:

- Jobs that access a particular database on a specific server must have a defined a set of environment variables before they can run and access the database correctly.
- Some jobs launched on a specific server must run with a specific user ID, and this user's environment settings must be active before the job can be executed correctly.
- All jobs names starting with *PXP* and launched on a particular server must be followed by a custom command that sends e-mail to department PXP with output from PXP jobs.
- All jobs launched on a particular Microsoft Windows server with a Windows user account that starts with *db2* must be preceded with a custom command that runs a specific DB2 sql query.

Tivoli Workload Scheduler provides support for two files on UNIX (and Linux), jobmanrc and .jobmanrc (dot plus *jobmanrc*), and one file on Microsoft Windows, jobmanrc.cmd, which can be used to manage, control, and tailor the job-scheduling environment on a workstation.

You can use the jobmanrc and .jobmanrc files to perform preprocessing as well as post-processing actions for jobs launched and executed by Tivoli Workload Scheduler.

4.2.1 The jobmanrc file

The jobmanrc (jobmanrc.cmd on Microsoft Windows) configuration script file is located in the TWShome directory and is used to specify or define job-scheduling environment settings effective for all jobs executed on the Tivoli Workload Scheduler workstation.

Because the jobmanrc file is called for all jobs (just before they are executed) on the workstation, the jobmanrc configuration file is considered a *global* configuration file (in effect for all jobs on the workstation).

The jobmanrc script sets variables used to locally configure the way jobs are launched on the workstation, regardless of the user.

The jobmanrc file is called by the Tivoli Workload Scheduler jobman process to launch a job. Jobman sets and exports a number of variables on the Tivoli Workload Scheduler workstation just before it calls the jobmanrc file. These job environment variables can be used in the jobmanrc file and in the job launched by jobmanrc.

The following are examples of job environment variables set by the Tivoli Workload Scheduler jobman process on Microsoft Windows workstations:

- HOMEPATH: Value of the HOMEPATH set in the user environment
- UNISON_HOST: Name of the host CPU (workstation)
- USERNAME: Value of the USERNAME set in the user environment

The following are examples of job environment variables set by the Tivoli Workload Scheduler jobman process on UNIX (and Linux) workstations:

- HOME: Home directory for the user
- LOGNAME: Logon user name
- UNISON_HOST: Name of the host CPU (workstation)

See the *IBM Tivoli Workload Scheduler Reference Guide V8.4,* SC32-1274, for a full list of job environment variables for UNIX and Windows.

4.2.2 The .jobmanrc file on UNIX workstations

On UNIX (and Linux) workstations, the local configuration script .jobmanrc permits users to establish a desired environment when processing their jobs. Unlike the jobmanrc script, the .jobmanrc script can be customized to perform different actions for different users.

Each user defined as TWSuser can customize in the home directory the .jobmanrc script to perform preprocessing and post-processing actions. Customizing the .jobmanrc script is an extra step that occurs before the job is actually launched (and also assumes control after the job has completed).

Jobs are not automatically run, so the command or script must be launched from inside the .jobmanrc.

If you are going to use a local configuration script, it must, at a minimum, run the job script file or the job command string ($UNISON_JCL).

All the variables exported into jobmanrc are available in the .jobmanrc shell. However, variables that are defined but not exported are not available.

Example 4-2 shows how to run a job's script file or command in a local configuration script.

Example 4-2 Simple .jobmanrc file that runs the job's script file or command

```
#!/bin/ksh
PATH=TWShome:TWShome/bin:$PATH
export PATH
/bin/sh -c "$UNISON_JCL"
```

4.3 Configuring Tivoli Workload Scheduler security

Security in Tivoli Workload Scheduler is accomplished with the use of a security file that contains one or more user definitions. Each user definition identifies a set of users, the objects they are permitted to access, and the types of actions they can perform.

Each time a Tivoli Workload Scheduler command is invoked using the Tivoli Workload Scheduler command-line interface or when the Tivoli Workload Scheduler is accessed from the Job Scheduling Console, security information is read from the security file to determine user access rights and defined capabilities. This file contains one or more *user definitions*. A user definition defines a group of one or more users who are either allowed or denied permission to perform specific actions against specific scheduling object types in the Tivoli Workload Scheduler database or plan.

The main Tivoli Workload Scheduler user, TWSuser, is defined in the security file. The TWSuser is defined together with the system administrator (root user on UNIX and Linux or administrator on Microsoft Windows) at installation. These users are the only users defined and allowed to connect to the user interfaces and to perform all operations on all scheduling resources.

Each workstation in a Tivoli Workload Scheduler network has its own security file. An individual file can be maintained on each workstation, or a single security file can be created on the master domain manager (MDM) and copied to each domain manager, fault-tolerant agent (FTA), and standard agent.

Using a single security file created on the MDM is referred to as *centralized security*. With centralized security, the security file for all workstations can be created, updated, or modified only on the MDM. The centralized security file must be distributed from the domain manager to all workstations in the Tivoli Workload Scheduler network after it is created on the MDM.

Note: Using the centralized security model provides a high degree of security in the Tivoli Workload Scheduler network or environment. On the other hand, centralized security is more complicated to maintain.

For most Tivoli Workload Scheduler installations, you must configure or tailor the security file to the environment where Tivoli Workload Scheduler is installed. You usually tailor the security file in situations where you use the security file as follows:

- To enforce naming conventions for scheduling objects
- To make sure that different groups of planners can access scheduling objects only for their groups
- To distinguish between different types of Tivoli Workload Scheduler users such as planners, operators, or administrators

Different types of Tivoli Workload Scheduler users normally have different access requirements to Tivoli Workload Scheduler.

4.3.1 Configuring the security file

The security file controls the following activities:

- Linking workstations
- Accessing command-line interface programs, the Job Scheduling Console, and the Tivoli Dynamic Workload Console
- Performing operations on scheduling objects in the database or in the plan

Each time a user runs Tivoli Workload Scheduler programs, commands, and user interfaces, the product compares the name of the user with the user definitions in the security file to determine if the user has permission to perform those activities on the specified scheduling objects.

The security file specifies which scheduling objects a user can manage and how. The settings are defined by writing user definitions. A user definition is an association between a name and a set of users, the objects they can access, and the actions they can perform on the specified objects.

When a user tries to access Tivoli Workload Scheduler either from the command-line or from the Job Scheduling Console, Tivoli Workload Scheduler first checks whether the user is defined in the security file.

If the user is defined in the security file, Tivoli Workload Scheduler checks whether the user has access to the object the user is trying to work with. Finally, if the user has access to the object, Tivoli Workload Scheduler checks whether the user has the requested access (such as modify, change, and update) to the object.

This three-level or three-tier security approach is shown in Figure 4-2.

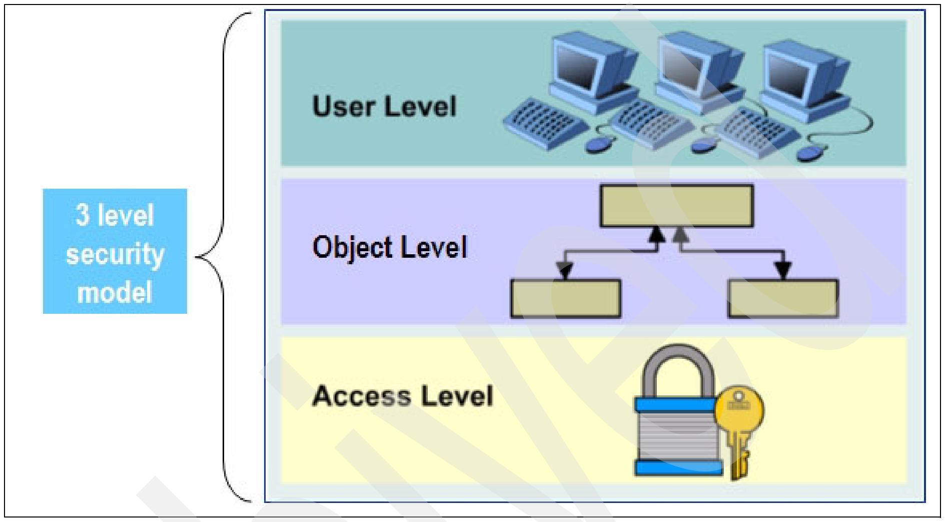


Figure 4-2 Three-level security model approach in Tivoli Workload Scheduler

The syntax in the Tivoli Workload Scheduler security file is as shown in Example 4-3.

Example 4-3 Tivoli Workload Scheduler security file syntax

```
[# comment]
user def-name user-attributes
begin [* comment]
object-type [object-attributes] access[=action[,...]] [object-type ...
]
[end]
```

In Example 4-3 you can see that each user definition can be divided into three levels or three parts, each of which is checked in the following order:

1. The user's details (*def-name* and *user-attributes*)
2. A list of the objects to which the user has rights (*object-type* and *object-attributes*)
3. The type of access the user has to these objects (*action*)

Note: The valid user types that can be specified in the Tivoli Workload Scheduler security file are: UNIX users, Microsoft Windows users, and LDAP users.

A security file installed together with Tivoli Workload Scheduler on a UNIX server for TWSuser, tws840, is shown in Example 4-4.

Example 4-4 Security file installed on UNIX server for user tws840

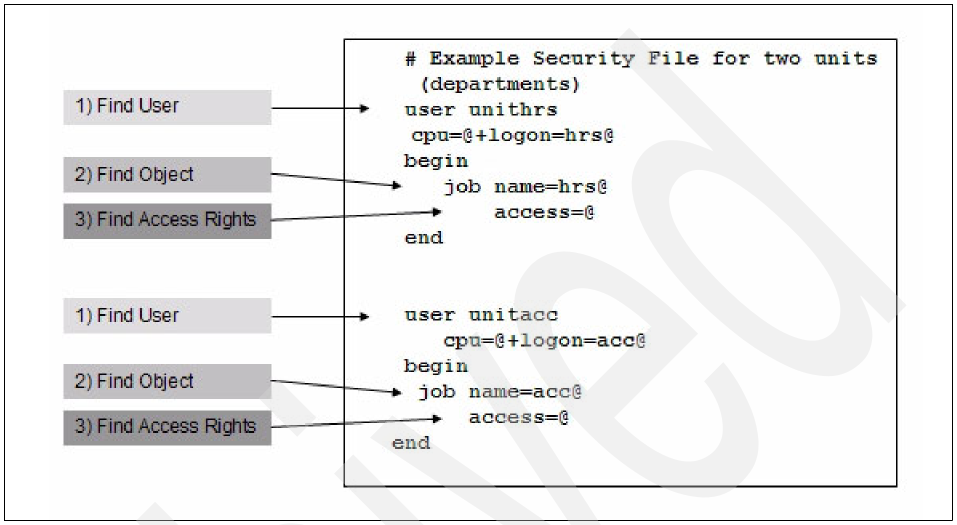
```
USER MAESTRO
    CPU=@+LOGON=tws840,root
BEGIN
    USEROBJ   CPU=@  ACCESS=ADD,DELETE,DISPLAY,MODIFY,ALTPASS,UNLOCK
    JOB       CPU=@  ACCESS=ADD,ADDDEP,ALTPRI,CANCEL,CONFIRM,DELDEP,
                        DELETE,DISPLAY,KILL,MODIFY,RELEASE,REPLY,RERUN,
                        SUBMIT,USE,LIST,UNLOCK
    SCHEDULE  CPU=@  ACCESS=ADD,ADDDEP,ALTPRI,CANCEL,DELDEP,DELETE,
                        DISPLAY,LIMIT,MODIFY,RELEASE,REPLY,SUBMIT,LIST,
                        UNLOCK
    RESOURCE  CPU=@  ACCESS=ADD,DELETE,DISPLAY,MODIFY,RESOURCE,USE,
                        LIST,UNLOCK
    PROMPT           ACCESS=ADD,DELETE,DISPLAY,MODIFY,REPLY,USE,LIST,
                        UNLOCK
    FILE      NAME=@ ACCESS=BUILD,DELETE,DISPLAY,MODIFY,UNLOCK
    CPU       CPU=@  ACCESS=ADD,CONSOLE,DELETE,DISPLAY,FENCE,LIMIT,
                        LINK,MODIFY,SHUTDOWN,START,STOP,UNLINK,LIST,
                        UNLOCK
    PARAMETER CPU=@  ACCESS=ADD,DELETE,DISPLAY,MODIFY,UNLOCK
    CALENDAR         ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,UNLOCK
    REPORT    NAME=@ ACCESS=DISPLAY
    EVENTRULE NAME=@ ACCESS=ADD,DELETE,DISPLAY,MODIFY,LIST,UNLOCK
    ACTION    PROVIDER=@  ACCESS=DISPLAY,SUBMIT,USE,LIST
    EVENT     PROVIDER=@  ACCESS=USE
END
```

The security file defines access to specified objects for selected users. The definition is for access to the Tivoli Workload Scheduler database through the composer command-line interface or the Job Scheduling Console and to the Tivoli Workload Scheduler plan through the conman command-line interface and the Job Scheduling Console.

You should tailor this preinstalled security file to specific Tivoli Workload Scheduler environment settings. You can perform this tailoring in a number of different ways using a number of different object types and object attributes to provide different levels of granularity in accessing Tivoli Workload Scheduler.

Figure 4-3 on page 61 shows a simple security file for two units or departments: the hrs (human resource) and the acc (accounting) departments. All users for the two departments can be identified from their user IDs that start with either hrs or acc.

If a user acc001 from the accounting department tries to access Tivoli Workload Scheduler, the user's access is checked by first finding an entry for the acc001 user. Then user access to objects are checked. In Figure 4-3, you can see that the acc001 user has full access (access=@) to all jobs starting with acc.



```
# Example Security File for two units
 (departments)
user unithrs
 cpu=@+logon=hrs@
begin
    job name=hrs@
        access=@
end

user unitacc
    cpu=@+logon=acc@
begin
  job name=acc@
      access=@
end
```

Figure 4-3 Simple security file for two units (departments)

In Figure 4-3 you can see it is possible to limit access in Tivoli Workload Scheduler, for example, to different departments (units) in an enterprise. If the enterprise has several departments and each department has several employees, you must create and enforce a naming convention based on departments in the Tivoli Workload Scheduler security file.

Furthermore, instead of defining access based on specific user IDs or names, you can group the users based on departments, and you can define access on the group level instead of the specific user level. This approach also makes management of the security file easier (because you do not have to change the security file every time a user leaves the company or changes to another department).

Note: For an user to access scheduling objects in the relational database (DB2 or Oracle) from the Tivoli Workload Scheduler composer command-line interface or the Job Scheduling Console, the user must exist only in the security file defined with sufficient access to the scheduling objects.

Important: The security file is checked sequentially from top to bottom to find matches for users, objects, and access rights, and Tivoli Workload Scheduler stops further checking when the first match is found.

Because the first matching statement is used, the order of object statements is important. They must be ordered from the most specific to the least specific (see Example 4-5).

Example 4-5 Incorrect and correct order for object statements

```
#Incorrect:
job name=@ access=display
job name=acc@ access=@

#Correct:
job name=acc@ access=@
job name=@ access=display
```

For further information about the Tivoli Workload Scheduler security file, refer to the *IBM Tivoli Workload Scheduler Reference Guide V8.4*, SC32-1274.

4.3.2 Updating the security file

The active or installed Tivoli Workload Scheduler security file is located in a compiled (machine-readable) file named security. This file resides in the TWShome directory.

Every workstation in a Tivoli Workload Scheduler network (domain managers, FTAs, and standard agents) has its own security file. It can be maintained on each workstation, or, if centralized security management is enabled, a security file can be created on the MDM and copied to each domain manager, FTA, and standard agent. You must ensure that all Tivoli Workload Scheduler users are assigned the required authorization in the security file.

The activation of the new security definitions does not require that you stop and restart either the Tivoli Workload Scheduler processes or the WebSphere Application Server infrastructure.

To modify the security file and activate the modified version, perform the following steps:

1. Run the **dumpsec** command to send information contained in the compiled security file to a specific text file:

   For example: `dumpsec > security.txt`

This command creates a text file, security.txt with a copy of the active security file definitions.

2. Modify the contents of the dumped security file to tailor it to the required user access specification.
3. Run **makesec** to install the security file and apply the modifications.

   For example: `makesec security.txt`

   This command compiles security definitions in the security.txt file and installs the security file in *TWShome*/Security.

   Changes to the security file are recognized as soon as conman or composer commands are issued again.

The process to be followed to modify or update the security file and apply the changes to the active security file is shown in Figure 4-4.

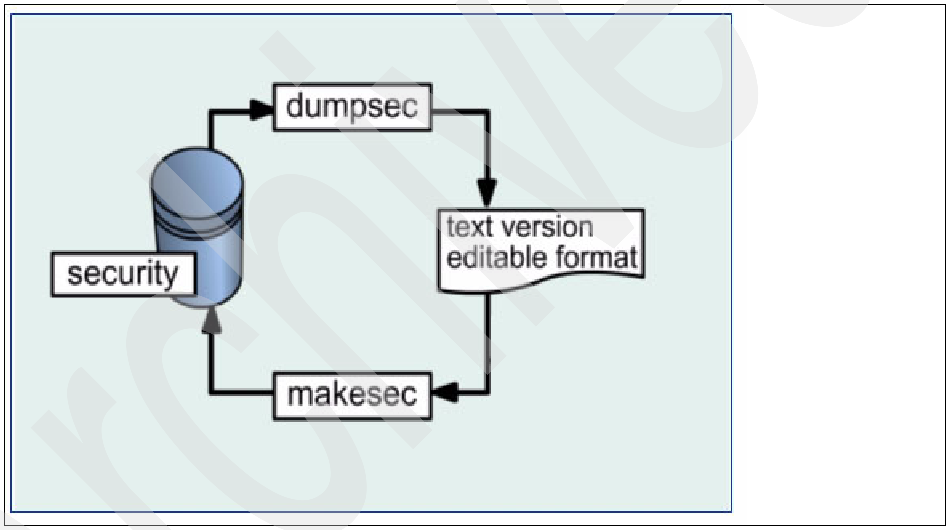


Figure 4-4 Process for modifying or updating the active security file

4.4 Configuring global and local options

After installing Tivoli Workload Scheduler, you may have to perform extra or optional configuration or customization of the installed Tivoli Workload Scheduler environment - for example:

- Tuning performance to deal with a growing number of jobs

  When Tivoli Workload Scheduler has been running for some time in the production environment, the number of jobs to be scheduled may have grown, possibly on a limited number of workstations, and you may have to tune performance.
- Changing the IP port number used for communication in the Tivoli Workload Scheduler network because of network reconfiguration.
- Implementing or using some functions in Tivoli Workload Scheduler that were not activated when Tivoli Workload Scheduler was first deployed in production.

In Tivoli Workload Scheduler, you can use two configuration files or options files to tailor Tivoli Workload Scheduler to specific environment settings and to tune Tivoli Workload Scheduler behavior and performance. These files are a *global* configuration file and a *local* configuration file.

4.4.1 The global configuration file

The global configuration file resides in the Tivoli Workload Scheduler relational database on the master domain manager, and the parameters in this file can be viewed, changed, and modified by the **optman** command.

As the name of the file implies, the global configuration file defines global settings that affect the entire Tivoli Workload Scheduler scheduling network. For example, startOfDay defines the start time for the Tivoli Workload Scheduler production plan.

To view the values of the global options file, use the following command:

```
optman ls
```

Output from **optman ls** on a test AIX MDM server is shown in Example 4-6.

Example 4-6 Output from optman ls command issued on a test AIX MDM

```
TECServerName / th = localhost
TECServerPort / tp = 5529
baseRecPrompt / bp = 1000
carryStates / cs =
companyName / cn = IBM Danmark
deploymentFrequency / df = 5
enCFInterNetworkDeps / ci = YES
enCFResourceQuantity / rq = YES
enCarryForward / cf = ALL
enCentSec / ts = NO
enDbAudit / da = 0
enEmptySchedsAreSucc / es = NO
enEventDrivenWorkloadAutomation / ed = YES
enEventProcessorHttpsProtocol / eh = YES
enLegacyId / li = NO
enLegacyStartOfDayEvaluation / le = NO
enListSecChk / sc = NO
enLogonBatch / lb = NO
enPlanAudit / pa = 1
enPreventStart / ps = YES
enRetainNameOnRerunFrom / rr = NO
enStrEncrypt / se = NO
enSwFaultTol / sw = NO
enTimeZone / tz = NO
eventProcessorEIFPort / ee = 33120
extRecPrompt / xp = 1000
ignoreCals / ic = NO
logCleanupFrequency / lc = 5
logHistory / lh = 10
logmanMinMaxPolicy / lm = BOTH
logmanSmoothPolicy / lt = -1
longDurationThreshold / ld = 150
mailSenderName / ms = TWS
maxLen / xl = 14
minLen / ml = 8
smtpServerName / sn = localhost
smtpServerPort / sp = 25
smtpUseAuthentication / ua = NO
smtpUseSSL / us = NO
smtpUseTLS / tl = NO
smtpUserName / un = tws840
smtpUserPassword / up = ****
startOfDay / sd = 0600
statsHistory / sh = 10
```

To list information about an option, use the following command:

```
optman show {optionName | optionShortName}
```

where `optionName` or `optionShortName` are options to be viewed

To change the values of an option, use the following command:

```
optman chg {optionName | optionShortName} = value
```

where `optionName` or `optionShortName` are options to be changed

All the option names can be seen in Example 4-7. The first name in the list is the optionName and the name after the forward slash (/) is the optionShortName.

For example, to view the value for the startOfDay option, simply enter:

```
optman show startOfDay
```

The output is shown in Example 4-7. The output following the value for the option lists a description of the option.

Example 4-7 Output from optman show startOfDay command

```
startOfDay / sd =0600
 Description:
Start time of day
Specifies the start time of the Tivoli Workload Scheduler
processing day in 24 hour format: "hhmm" (0000-2359).
The default start time is "0600" (06:00 AM), and the default
launch time of the "final" job stream is "0559" (05:59 AM).
After changing this option, you must also change the
launch time of the "final" job stream, which is usually
set to one minute before the start time, and create a
new production plan to make the change effective.
```

Because the short name for the startOfDay option is sd, the options can also be listed or viewed by issuing the command:

```
optman show sd
```

For example, to view the value for smtpServerName, enter:

```
optman show smtpServerName
```

The output is shown in Example 4-8.

Example 4-8 Output from optman show smtpServerName command

```
smtpServerName / sn =localhost
 Description:
 SMTP server name
Specifies the name of the SMTP server used by the mail plug-in.
The default value is "localhost".
```

```
After changing the value of this parameter, the change is
effective for the next mail send action performed.
```

Note: The smtpServerName parameter is used to configure the mail server for event-based jobs.

The smtpServerName option specifies the host name or IP address of the SMTP (Simple Mail Transfer Protocol) server through which outgoing e-mails are delivered.

As can be seen from Example 4-6 on page 64, you can change and configure many different parameters or options in the global configuration file.

Several of the global options are new and are introduced in Tivoli Workload Scheduler Version 8.4. These new global options are primarily related to the new event-driven workload automation feature introduced in Tivoli Workload Scheduler Version 8.4.

The global options parameters are described or detailed further here. For a detailed description of the global options, refer to *IBM Tivoli Workload Scheduler Planning and Installation Guide v8.4*, SC32-1273.

4.4.2 The local configuration file

The local configuration file, localopts, resides in the TWShome directory on each Tivoli Workload Scheduler workstation (or agent) in the Tivoli Workload Scheduler network.

As the name of the file implies, the localopts configuration file defines local settings for the Tivoli Workload Scheduler workstation. Settings in the localopts file affect only the specific Tivoli Workload Scheduler workstation.

When the Tivoli Workload Scheduler workstation is installed on a server, a working copy of the localopts file is placed in *TWShome*/localopts. This localopts file is preconfigured with settings specified during the Tivoli Workload Scheduler installation process.

Changes in the localopts file do not take effect until Tivoli Workload Scheduler is stopped and restarted on the server.

The localopts file contains many configuration options, some of are described in "Localopts options you can change after installation" on page 68.

For many Tivoli Workload Scheduler workstations and installations, you do not have to change or tailor the localopts file after the Tivoli Workload Scheduler has been installed on the server. The default settings are sufficient for most Tivoli Workload Scheduler installations.

For a full list and description of all the localopts options, refer to *IBM Tivoli Workload Scheduler Planning and Installation Guide v8.4*, SC32-1273.

Localopts options you can change after installation

In some situations you can change the localopts options after installing Tivoli Workload Scheduler on a server. For example, these are circumstances in which you may want to change the localopts options:

- The number of jobs to be handled on the installed Tivoli Workload Scheduler workstation has increased considerably, and throughput and response time as well as performance for the Tivoli Workload Scheduler workstation have become an issue.
- The port number to be used by the Tivoli Workload Scheduler workstation was specified incorrectly during installation or must be changed due to changed network configuration.
- When a job or job stream is late, you can configure job deadline notification.

Localopts configuration related to performance

Tivoli Workload Scheduler in general is an I/O intensive application. Information is passed by events between the different Tivoli Workload Scheduler processes, and some of the Tivoli Workload Scheduler processes - for example, mailman - actually can perform a high number of I/Os (disk read and writes) when the number of jobs on the Tivoli Workload Scheduler workstation is high.

If I/O throughput or I/O response time starts to be a problem for the Tivoli Workload Scheduler workstation, consider tuning the following localopts options:

- Mailman cache

  localopts options: `mm cache mailbox and mm cache size`

  The default setting for mm cache mailbox is No, which means that the memory cache is not used when events are written to disk.

  Because using the cache is much faster than writing directly to disk, changing mm cache mailbox to Yes can reduce I/O considerably for a busy Tivoli Workload Scheduler workstation.

  When you change mm cache mailbox to Yes, you must also consider whether the size of the cache specified in mm cache size is adequate. While the default value 32 bytes is a reasonable value for most small and mid-sized Tivoli Workload Scheduler installations, you can extend it up to 512 bytes.

- File system synchronization level (UNIX and Linux only).

  Localopts option: `sync level`

  The default setting for sync level is low. Possible values are low, medium, or high.

  The sync level attribute specifies the frequency at which Tivoli Workload Scheduler synchronizes messages held on disk with those in memory.

  The default value low enables the operating system to handle the speed of write accesses. This option speeds up all processes that use mailbox files. Disk usage is notably reduced. If the file system is reliable, data integrity is assured anyway.

  Consider changing this option to medium or high only if you have old technology disks on your system and want to avoid any possible loss of data. However, keep in mind that changing from the default value low to either medium or high increases the number of I/Os and decreases I/O throughput.

Another problem with performance can be the time used to distribute a new Symphony file to a Tivoli Workload Scheduler workstation (or FTA), especially if this workstation must be reached over a slow or unstable network connection.

To reduce the size of the Sinfonia file sent to the Tivoli Workload Scheduler workstation, consider using the Sinfonia compression feature in Tivoli Workload Scheduler. Use of the compression feature can be particularly helpful when the Symphony file is large and the network connection between two nodes is slow or unreliable (for example, in a WAN).

The following setting in localopts is used to set the compression in Tivoli Workload Scheduler:

```
wr enable compression=yes
```

This setting means that Sinfonia is compressed before being sent to the Tivoli Workload Scheduler FTA.

Note: Due to the overhead of compression and decompression, we recommend you use compression if the Sinfonia file is 4 MB or larger.

Port number used by the Tivoli Workload Scheduler workstation

The IP port number used by the Tivoli Workload Scheduler workstation when establishing a connection to its master domain manager or domain manager is important. If the Tivoli Workload Scheduler workstation does not listen on the expected port number, it is not possible to establish a two-way network connection between the Tivoli Workload Scheduler workstation and its domain manager.

Installation of the Tivoli Workload Scheduler workstation specifies the port number that the workstation should use. This port number is added to the localopts file in the nm port option.

When netman is started on this Tivoli Workload Scheduler workstation, the workstation starts to listen on the nm port. The port in localopts nm port must match the port specified for the workstation defined in the Tivoli Workload Scheduler MDM database, using the TCPADDR keyword (see 4.1.3, "Configuring an agent or domain manager workstation" on page 50).

If for any reason you have to change the port, do so by changing the port number in the localopts nm port option and in the corresponding workstation definition in the Tivoli Workload Scheduler MDM database.

Configuring Tivoli Workload Scheduler to check for late jobs

If deadlines are used for job streams or jobs in Tivoli Workload Scheduler, it may be necessary to configure Tivoli Workload Scheduler to receive notification when a job or job stream is *late* - that is, the job or job stream has passed its deadline and has not yet started or completed.

Jobs or job streams that have not yet started or are still running when the deadline is reached are considered late in the plan. When a job (or job stream) is late, the following actions are performed:

1. The job is shown as late in conman and the Job Scheduling Console.
2. An event is sent to the IBM Tivoli Enterprise Console and the IBM Tivoli Business Systems Manager (optional).
3. A message is issued to the stdlist and console logs.

Note: To enable checking for late jobs in Tivoli Workload Scheduler and to enable Tivoli Workload Scheduler to check for the deadline keyword, the bm check deadline variable must be set to a value higher than 0 in the localopts configuration file on the target Tivoli Workload Scheduler workstation for the job or job stream. The default value for bm check deadline is 0.

4.5 Configuring extended agents

Workstations are normally physical assets (that is, computers), but they can also be logical definitions hosted by a physical workstation, representing operating systems or applications where Tivoli Workload Scheduler should run jobs or job streams. In this case, they are defined as *extended agents*.

An extended agent (XA) is defined as a workstation that has a host and an access method. Extended agents are used to extend the job-scheduling functions of Tivoli Workload Scheduler to other systems and applications. The host is any other workstation, except another extended agent. The access method is an IBM-supplied or user-supplied script or program run by the host whenever the extended agent is referenced in the production plan. For example, to launch a job on an extended agent, the host runs the access method, passing it job details as command-line options. The access method communicates with the external system or application to launch the job and return the status of the job.

Workstation definition for an extended agent

Each extended agent must have a logical workstation definition. This logical workstation must be hosted by a Tivoli Workload Scheduler physical workstation, either a master domain manager, domain manager, or FTA workstation. The extended agent workstation definition references the name of the access method and the host workstation. When jobs are launched on the extended agent workstation, the access method is called and passes the job information to the external system.

When viewing an extended agent in the Tivoli Workload Scheduler plan, the *linked* information for agent workstations does not imply that an IP connection exists between the host workstation and the extended agent workstation. Instead, the linked information is an indication showing that it is possible to schedule jobs on the extended agent workstation. The IP connection or IP link is between the Tivoli Workload Scheduler manager workstation and the extended agent host workstation.

Example 4-9 shows how an extended agent workstation can be defined in Tivoli Workload Scheduler, using the composer command-line interface. The extended agent workstation in the example uses the unixlocl access method. See 4.5.2, “Extended agent types” on page 74 for a description of this method.

Example 4-9 Definition for XA workstation using unixlocl access method

```
CPUNAME UNIXLOCAL
  DESCRIPTION "Extend Agent workstation for local UNIX workstation"
  OS OTHER
  NODE null TCPADDR 11111
  FOR MAESTRO HOST $MASTER ACCESS "unixlocl"
    TYPE X-AGENT
    AUTOLINK OFF
    BEHINDFIREWALL OFF
    FULLSTATUS OFF
END
```

The NODE keyword must be null for the extended agent workstation, and ACCESS must be set as unixlocl to run the unixlocl access method.

Note: The HOST keyword contains a $MASTER variable. The $MASTER specification means that the extended agent workstation is to be hosted by the active master domain manager.

Depending on the setting for the localopts mm resolve master option, the $MASTER variable is resolved either when the plan is extended (mm resolve master = no) or when a **switchmgr** command is issued (mm resolve master = yes).

Use of the $MASTER keyword in the extended agent workstation definition makes it possible to continue job scheduling on the extended agent workstation after a switch is performed to the backup master domain manager because the agent is hosted by the backup master domain manager after the switch.

4.5.1 Extended agent attributes and job execution process

Figure 4-5 shows the relationship between the extended agent's host and access methods.

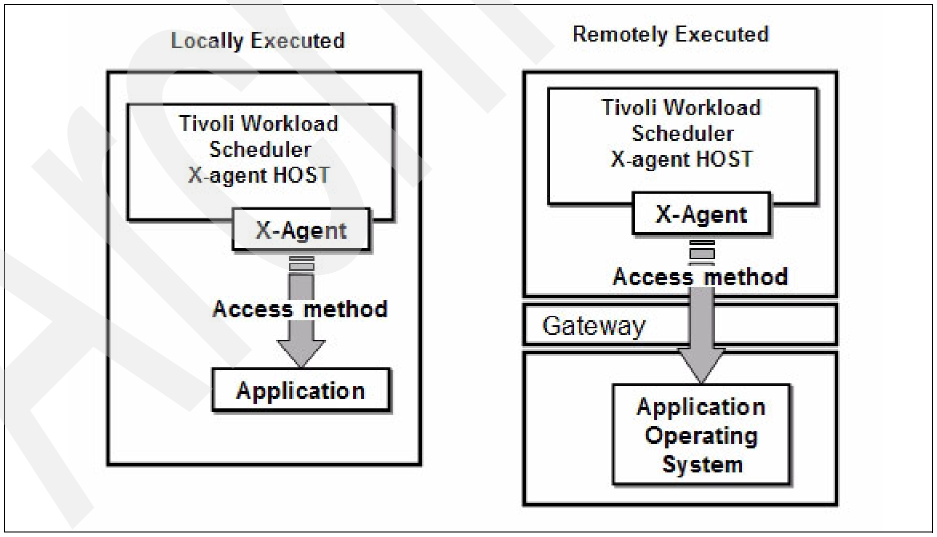

Locally Executed
Tivoli Workload Scheduler X-agent HOST
X-Agent
Access method
Application
Remotely Executed
Tivoli Workload Scheduler X-agent HOST
X-Agent
Access method
Gateway
Application Operating System

Figure 4-5 Tivoli Workload Scheduler extended agent and its attributes

The following lists provides a detailed explanation of Figure 4-5 on page 72:

- Tivoli Workload Scheduler X-agent HOST

  The X-agent's HOST is the extended agent's host in the Tivoli Workload Scheduler workstation that performs the scheduling functions for the extended agent. The extended agent can be hosted by any type of Tivoli Workload Scheduler workstation except another extended agent

- Access method

  The access method is supplied, for example, by IBM, Tivoli Workload Scheduler, or a user-created script or program executed on a Tivoli Workload Scheduler host whenever the extended agent is referenced in the production plan (in a Tivoli Workload Scheduler job). For example, to launch a job on an extended agent, Tivoli Workload Scheduler executes the access method, passing it job details, such as the job name, script name, and user name, as command-line arguments. The access method communicates with the external system or application to launch the job and return the status to Tivoli Workload Scheduler.

- Locally Executed

  On a locally executed extended agent, the Tivoli Workload Scheduler host and the extended agent reside on the same machine or server. The access method translates the job information, passes it to the application, and awaits a return code.

- Remotely Executed

  On a remotely executed extended agent, the Tivoli Workload Scheduler host workstation and the extended agent reside on different machines. In this case, the extended agent's access method provides the networking capability to connect to the foreign host and communicate the job information to the agent. The access method then awaits the return of an exit code or periodically connects to the foreign host to check on the job status.

In Figure 4-6 on page 74, the job execution process on the extended agent workstation is shown. The methods.opts file contains configuration information used to access the application or operating system through the access method.

The configuration information can be the user who accesses the application, connect information (IP address information), and so on. The content of the methods.opts configuration file depends on which application or operating system the access method is going to access.

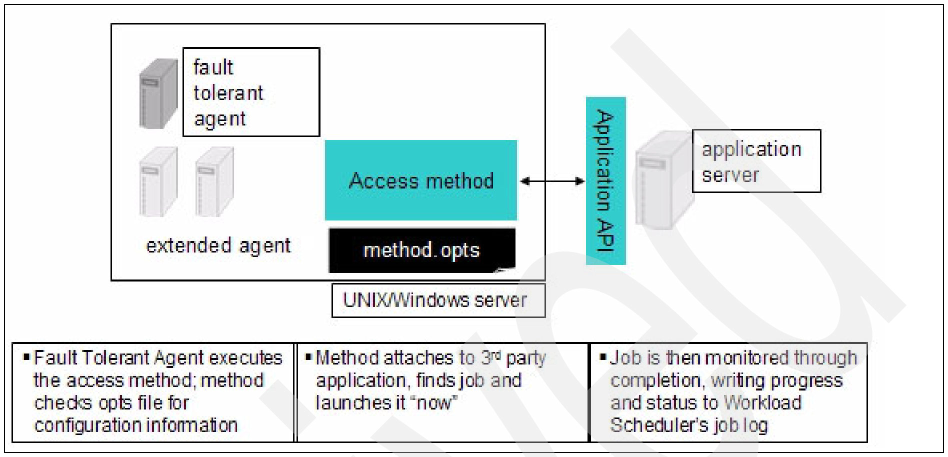

fault tolerant agent
extended agent
Access method
method.opts
UNIX/Windows server
Application API
application server
▪ Fault Tolerant Agent executes the access method; method checks opts file for configuration information
▪ Method attaches to 3rd party application, finds job and launches it "now"
▪ Job is then monitored through completion, writing progress and status to Workload Scheduler's job log

Figure 4-6 Job execution on extended agent workstation

4.5.2 Extended agent types

The three types of extended agent are as follows:

- Extended agent for other operating systems
- Extended agent for application processing
- Extended agent for inter-networking

Extended agents for other operating systems

Tivoli Workload Scheduler includes access methods for two types of UNIX extended agents, installed during Tivoli Workload Scheduler installation.

The local UNIX access method is used to enable a single UNIX workstation to operate as multiple Tivoli Workload Scheduler workstations, all of which can run Tivoli Workload Scheduler scheduled jobs. The remote UNIX access method designates a remote UNIX workstation, which does not have Tivoli Workload Scheduler installed on it, to run Tivoli Workload Scheduler scheduled jobs.

Note: In addition to these two extended agents for UNIX, it is possible to purchase third-party extended agents - for example, for scheduling on OpenVMS operating systems.

Local UNIX access method

The local UNIX access method can be used to define multiple Tivoli Workload Scheduler workstations on one workstation: the host workstation and one or more extended agents. When Tivoli Workload Scheduler sends a job to a local UNIX extended agent, the access method, unixlocl (installed in the *TWShome*/methods directory) is invoked by the host to run the job. The method starts by running the standard configuration script on the host workstation (*TWShome*/jobmanrc). If the job's logon user is permitted to use a local configuration script and the script exists as $HOME/.jobmanrc, the local configuration script is also run. The job itself is then run by either the standard or the local configuration script. If neither configuration script exists, the method starts the job.

The launching of the configuration scripts, jobmanrc and .jobmanrc (see 4.4, "Configuring global and local options" on page 64) can be configured in the method script. The method runs the configuration scripts by default if they exist. To disable this feature, you must comment out a number of lines in the method script. For more information, examine the script file *TWShome*/methods/unixlocl the extended agent's UNIX host.

Remote UNIX access method

The remote UNIX access method can be used to designate a non-Tivoli Workload Scheduler workstation to run jobs scheduled by Tivoli Workload Scheduler. It is possible to use unixrsh or unixssh:

- The unixrsh method

  When Tivoli Workload Scheduler sends a job to a remote UNIX extended agent, the access method, unixrsh, creates a /tmp/maestro directory on the non-Tivoli Workload Scheduler workstation. It then transfers a wrapper script to the directory and runs it. The wrapper then runs the scheduled job. The wrapper is created only once, unless it is deleted or moved, or is outdated.

  To run jobs using the extended agent, the job logon users must be given appropriate access on the non-Tivoli Workload Scheduler UNIX workstation. To do this, a .rhost, /etc/host.equiv file, or its equivalent, must be set up on the workstation. If open file dependencies are to be checked, root access must also be permitted. For more information about the access method, examine the script file *TWShome*/methods/unixrsh on an extended agent's host.

- The unixssh method

  The unixssh method works like unixrsh but uses a secure remote shell to connect to the remote host. The files used by this method are *TWShome/*methods/unixssh and *TWShome/*methods/unixssh.wrp.

  The unixssh method uses the ssh keyword. It is possible to generate this keyword with any tools that are compatible with the secure remote shell.

Extended agents for application processing

Extended agents for application processing makes it possible for Tivoli Workload Scheduler to schedule jobs in third-party applications or products such as the following:

- Oracle E-Business Suite
- PeopleSoft
- SAP R/3
- z/OS

To schedule jobs in any of these applications or products, you must use an access method developed and programmed to perform the correct initialization with the application and make the correct call to the application. The access methods for these applications can be purchased with IBM Tivoli Workload Scheduler for Applications Version 8.4.

Note: In addition to the extended agent for applications contained in the Tivoli Workload Scheduler for Applications Version 8.4 package, Tivoli Workload Scheduler Version 8.4 is also shipped with an extended agent for Tivoli Storage Manager. This extended agent can be used to execute commands on the Tivoli Storage Manager server for administrative and client backup purposes.

The extended agent methods script file tsmxagent is installed in the *TWShome*/methods directory together with an example of the tsmxagent.opts options file.

The access methods are named as follows:

- MCMAGENT for Oracle E-Business Suite
- psagent for PeopleSoft
- r3batch for SAP R/3
- mscca7 for CA-7, mvsjes for JES2 and JES3, and mvsopc for OPC and Tivoli Workload Scheduler for z/OS

To launch a job in an external application, Tivoli Workload Scheduler runs the extended agent access method, providing it with the extended agent workstation name and with information about the job. The method checks the corresponding options file to determine which external environment instance it must connect to. The access method can then launch jobs on that instance and monitor them through completion, writing job progress and status information in the standard list file of the job.

Note: In addition to the access method, z/OS and SAP extended agents require a gateway component that is installed on the target z/OS and SAP applications, respectively. Similarly, the PeopleSoft extended agent requires a project component that is installed on the target PeopleSoft system. All these components are shipped with Tivoli Workload Scheduler for Applications. Refer to *IBM Tivoli Workload Scheduler for Applications User's Guide v8.4,* SC32-1278 for more details.

The Tivoli Workload Scheduler extended agent method program checks for two options files:

- The global options file, defining settings common to all extended agent workstations using the access method
- The WORKSTATIONNAME_accesmethod.opts file, defining configuration options specific to each extended agent workstation within a particular installation of an access method

  Typically, the WORKSTATIONNAME_accesmethod.opts file is used when more than one extended agent is going to be hosted by the same HOST workstation and each extended agent workstation requires specific configuration settings.

The following section includes examples of configuration files for the SAP R/3 extended agent workstation, and examples of the options that can be defined in the SAP R/3 extended agent options file. You can change these options files using the Option Editor, which is part of the Tivoli Workload Scheduler for Applications, or by using any text editor. Option Editor provides a GUI interface that enables you to easily change these option files.

For further details for the extended agents provided in Tivoli Workload Scheduler for Applications, refer to *IBM Tivoli Workload Scheduler for Applications User's Guide v8.4,* SC32-1278.

Tivoli Workload Scheduler for Applications SAP R/3 extended agent

Tivoli Workload Scheduler launches jobs in SAP using jobs defined to run on a Tivoli Workload Scheduler extended agent workstation. A SAP extended agent workstation is defined as a Tivoli Workload Scheduler workstation that is hosted by a domain manager, FTA, or standard agent workstation that uses the r3batch access method. The SAP extended agent workstation uses the access method r3batch to pass SAP job-specific information to predefined SAP instances. The access method uses information provided in an options file to connect and launch jobs on an SAP instance.

You can define multiple extended agent workstations to use the same host by using a combination of one global options file and multiple extended agent workstation specific options files. Using the SAP extended agent name as a key, r3batch uses the corresponding options file to determine which instance of SAP will run the job. It makes a copy of a template job in SAP and marks it to run with a start time of start immediate. It then monitors the job to completion, writing job progress and status information to a job standard list on the host workstation.

To be able to schedule jobs in and get job status from SAP, Tivoli Workload Scheduler for Applications provides correction and transport (containing SAP ABAP/4 modules) files that you must load in the SAP instance where Tivoli Workload Scheduler is going to schedule and run SAP jobs.

For each SAP extended agent workstation's configuration information, you must supply SAP instance information such as the following (these are the required option settings):

- r3client

  SAP R/3 client number
- r3host

  Host name of the SAP message server when using logon groups or, in all other cases, the host name of the application server
- r3instance

  SAP instance number
- r3password

  Password for the r3user. Ensure you enter the same password when creating this user in the SAP system. The password, which is stored in encrypted format, can be a maximum of eight characters. The value is case sensitive.
- r3sid

  SAP system ID
- r3user

  Name of the SAP user with which the access method connects to the SAP system. It must have the appropriate privileges for running background jobs. It is sometimes referred to as the Maestro™ User ID.

Because a one-to-one relationship exists between the SAP instance identified by these required options and the associated Tivoli Workload Scheduler extended workstation, you must define the preceding options in the local options (workstation-specific) file.

The options files used by the SAP access method, r3batch, are the following:

- r3batch.opts

  A common configuration file for the r3batch access method, the settings of which affect all the r3batch instances. It functions as a “global” configuration file.

- *WORKSTATIONNAME*_r3batch.opts

  A configuration file specific to each Tivoli Workload Scheduler extended agent workstation that uses the r3batch access method. Its options affect only the r3batch instance used by that particular workstation. It functions as a “local” configuration file.

For example, to define two extended agents, wkst1 and wkst2, that access two SAP systems, SAP1 and SAP2, with the r3batch access method, you must define the following configuration files:

- Global r3batch.opts
- Local file: WKST1_r3batch.opts
- Local file WKST2_r3batch.opts

The two workstations, wkst1 and wkst2, should be defined as extended agent workstations, using access method r3batch and hosted by a domain manager, FTA, or standard agent workstation. Because workstations wkst1 and wkst2 are hosted by the same workstation in Tivoli Workload Scheduler, the local files WKST1_r3batch.opts and WKST2_r3batch.opts are used to distinguish between different SAP application instances.

Note: If r3batch finds the local configuration file for an extended agent, it ignores the duplicate information contained in r3batch.opts file.

There are no minimum requirements for the global r3batch.opts file because all required options can be defined in the local workstation configuration file WORKSTATIONNAME_r3batch.opts.

Specifying global or common options in the r3batch.opts file is also referred to as *inheriting* configuration options. In other words, the extended agent workstation inherits options from the global r3batch.opts file. If the options are not overwritten in the local WORKSTATIONNAME_r3batch.opts options file, they are in effect on the extended agent workstation (that is, they have been inherited from the global r3batch.opts file).

Because the local options file contains the password for r3user defined in the options file, take care if you manually modify the local configuration file using a text editor.

When entries are added to or modified in the options file, using the Tivoli Workload Scheduler options editor program, opted.bin (or opeted.bat on Microsoft Windows), the password value is automatically encrypted before it is written in the file.

If you modify the option file with a text editor, you must run the enigma program to encrypt the password before writing it in the file.

To run the encryption program, enter the following command:

```
enigma [password]
```

The password can be included on the command line or entered in response to a prompt. The program returns an encrypted version that should be entered in the options file.

Extended agent for internetworking

Extended agents for internetworking are used to create Tivoli Workload Scheduler *internetwork dependencies*. The internetwork dependencies enable jobs and job streams in the local Tivoli Workload Scheduler network to use jobs and job streams in a remote Tivoli Workload Scheduler network as *follows dependencies*.By using follows dependencies, you can define the other jobs and job streams that must complete successfully before a job or job stream is launched.

Before you can define internetwork dependencies, you must create a workstation definition for the *network extended agent*. A network extended agent is a Tivoli Workload Scheduler workstation that handles follows dependencies between its local network and a remote Tivoli Workload Scheduler network.

The local Tivoli Workload Scheduler network can include more than one network agent, each representing a specific Tivoli Workload Scheduler remote network where jobs and job streams referring to locally defined internetwork dependencies are defined. Internetwork dependencies are assigned to jobs and job streams in the same way as local follows dependencies, with the exception that the network agent's name is included to identify the followed job or job stream.

A special job stream, EXTERNAL, is automatically created by Tivoli Workload Scheduler for each network agent in the local network. It contains placeholder jobs to represent each internetwork dependency.

An EXTERNAL job is created for each internetwork dependency belonging to job streams planned to start on different days with different schedule dates. This means that EXTERNAL jobs differ from another one by the following:

- The script file name, which identifies the remote job or job stream the local job or job stream is dependent on.
- The date the local job stream containing the internetwork dependency is planned to start. If the dependency is defined in a job within the job stream, the date the job stream is planned to start is taken into account.

The check of the internetwork dependency does not start until the job stream matches its time dependency or it is released.

Network extended agents are defined as extended agents and require a hosting physical workstation (UNIX, Linux, or Microsoft Windows) and an access method. netmth is the access method for network agents.

You must create an options file, netmth.opts, on the workstation where the network agent runs. In this file, you define the user under which the access method runs, and the time to wait to receive a response from the access methods before shutting down. The netmth.opts option file must have the same path as the access method:

```
TWShome/methods/netmth.opts
```

The content of the netmth.opts file has the following structure:

```
GSuser=login_name
GStimeout=seconds
```

where:

- *login_name*

  The logon used to run the method. If the network agent's host is a workstation running Microsoft Windows, the user must be defined in Tivoli Workload Scheduler.

- *seconds*

  The number of seconds Tivoli Workload Scheduler waits for a response before shutting down the access method. The default setting is 300 seconds. The next time batchman checks the status of the remote predecessor job or job stream, the access methods starts up automatically.

Changes to the netmth.opts file do not take effect until Tivoli Workload Scheduler is stopped and started again.

Figure 4-7 shows how an extended agent can be defined to make it possible to define internetwork dependencies in MasterB (a remote network) for jobs in MasterA (the local network).

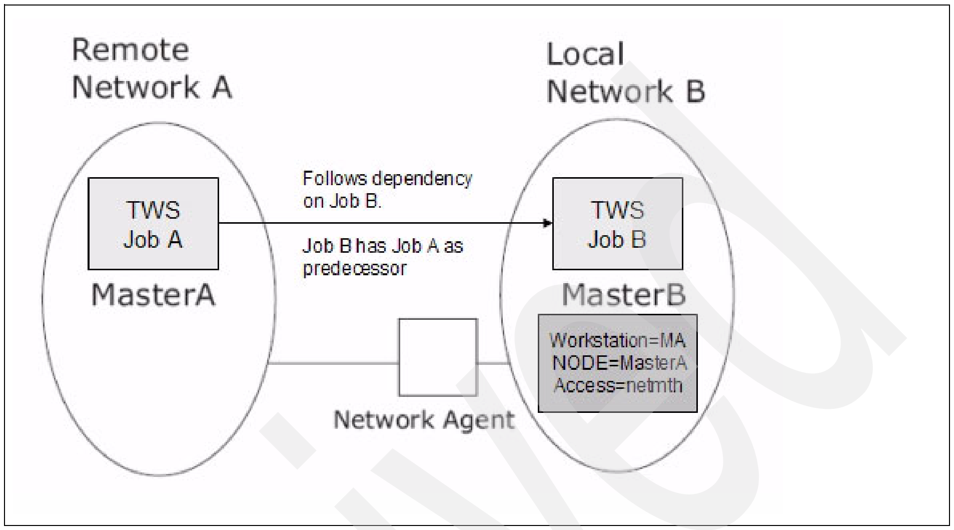


Figure 4-7 Extended agent workstation defined for internetwork dependencies

To make internetwork dependencies in MasterB for jobs in MasterA, a network agent workstation, named MA, is defined in MasterB to manage internetwork dependencies on jobs or job streams defined in Network A (in MasterA). The MA workstation is defined as an extended agent using access method netmth and the node address (DNS name manager), DNS name for the MasterA master domain manager. Host workstation for the MA workstation should be MasterB.

Using this MA network workstation, you can define Job B in Tivoli Workload Scheduler so that it has Job A in Tivoli Workload Scheduler master domain manager MasterA as predecessor.

MasterB checks the status of job A every *bm check status* number of seconds, specified in the *TWShome*/localopts file.

4.6 Configuring WebSphere Application Server

Connections between the Tivoli Workload Scheduler engine, the Job Scheduling Console, and the command line (**`composer`** and **`conman`** commands) requires the embedded version of IBM WebSphere Application Server V6.1. It is installed automatically when a master domain manager or backup master domain manager is installed. In this section, the embedded version of IBM WebSphere Application Server V6.1 is referred to as WebSphere Application Server.

Note: The embedded version of the WebSphere Application Server Version 6.1 is not the same as WebSphere Application Server - Express. The embedded version of IBM WebSphere Application Server V6.1 is a runtime version of WebSphere Application Server Version 6.1, which is bundled in and managed by Tivoli Workload Scheduler.

The WebSphere Application Server is configured and started during the installation of Tivoli Workload Scheduler Version 8.4. Settings for the WebSphere Application Server configuration are provided during installation either by the Tivoli Workload Scheduler installation wizard or by a response file if silent installation is performed. No post-installation steps are required for the WebSphere Application Server after Tivoli Workload Scheduler is installed. However, during the product life cycle, you may have to change or reconfigure some of the configuration parameters or settings for the WebSphere Application Server for a number of reasons, including the following:

- The port number used for the Job Scheduling Console communication has to be changed, for example, due to a redesign of the network.
- The TCP ports used by the WebSphere Application Server have to be customized to cross a firewall where Network Address Translation (NAT) is used.
- The password of TWSuser or DB2user has changed.
- The DB2 server connection information has changed.
- The user registry is moved on to a Lightweight Directory Access Protocol (LDAP).
- An application server trace is required by IBM support for problem determination purposes.

Examples are provided in the sections that follow to illustrate how you can change WebSphere Application Server parameters or settings to accomplish the required reconfiguration of WebSphere Application Server.

To facilitate the process of adjusting or changing parameter settings, the embedded version of WebSphere Application Server provides a set of utilities that can be used to configure WebSphere Application Server settings.

The utilities are a set of scripts based on Microsoft Windows batch files, UNIX and Linux shell scripts, and WebSphere Application Server Jacl procedures. The scripts run WebSphere Application Server embedded utilities that perform the reconfiguration. Many of them load configuration settings from properties files. Templates for these files are also provided.

Tivoli Workload Scheduler installs the following Microsoft Windows batch files:

- backupConfig.bat
- changeDataSourceProperties.bat
- changeHostProperties.bat
- changeSecurityProperties.bat
- changeTraceProperties.bat
- encryptProfileProperties.bat
- restoreConfig.bat
- showDataSourceProperties.bat
- showHostProperties.bat
- showSecurityProperties.bat
- startWas.bat
- stopWas.bat
- updateWasService.bat

Tivoli Workload Scheduler installs the following UNIX and Linux scripts:

- backupConfig.sh
- changeDataSourceProperties.sh
- changeHostProperties.sh
- changeSecurityProperties.sh
- changeTraceProperties.sh
- encryptProfileProperties.sh
- restoreConfig.sh
- showDataSourceProperties.sh
- showHostProperties.sh
- showSecurityProperties.sh
- startWas.sh
- stopWas.sh

The following templates are installed for both Microsoft Windows and UNIX (and Linux) operating systems:

- DataSourceProps.properties
- HostConfigProps.properties
- SecurityProps_FULL.properties

- SecurityProps_TEMPLATE.properties
- TracingProps.properties

All scripts and batch and template files are installed in *TWShome/*wastools*.*

4.6.1 Changing host, data source, or security properties

The name of the files or scripts enables you to determine the purpose of the scripts. Some scripts are normally used together in a process, where the first script is used to generate a text file with a copy of the WebSphere Application Server runtime settings, then to modify the text file with the required changes, and finally to activate the modified text file in WebSphere Application Server. See Figure 4-8 for an example.

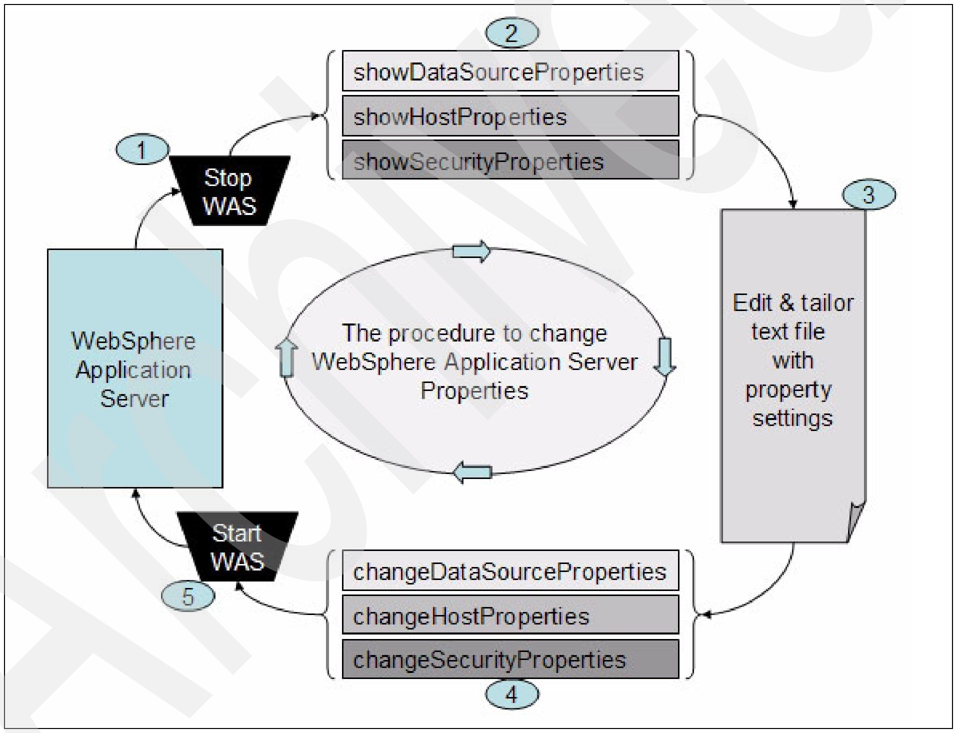

2
showDataSourceProperties
showHostProperties
showSecurityProperties
1
Stop WAS
3
Edit & tailor text file with property settings
WebSphere Application Server
The procedure to change WebSphere Application Server Properties
Start WAS
5
changeDataSourceProperties
changeHostProperties
changeSecurityProperties
4

Figure 4-8 Changing WebSphere Application Server properties

Before you make any changes to the WebSphere Application Server properties, we recommend you back up the WebSphere Application Server configuration using the scripts *TWShome*/wastools/backupConfig.sh (UNIX or Linux) and *TWShome*\wastools\backupConfig.bat (Microsoft Windows). Creating a backup enables you to restore the saved configuration at a later date, if required, using the *TWShome*/wastools/restoreConfig.sh script (UNIX or Linux) or *TWShome*\wastools\restoreConfig.bat script (Microsoft Windows).

If started with no arguments, backupConfig.sh creates a new file, which includes the current date - for example, WebSphereConfig_2008-05-28.zip - in the current directory.

Note: The backupConfig.sh script stops the WebSphere Application Server. Remember to restart WebSphere Application Server or wait until it is automatically restarted by the appserverman process.

For more information regarding the backup and restore scripts, see *IBM Tivoli Workload Scheduler Administration and Troubleshooting V8.4,* SC32-1275.

The following provides information about the show<*property_type*>Properties.sh and change<*property_type*>Properties.sh utilities shown in Figure 4-8 on page 85:

- ChangeDataSourceProperties

  Used to change the data source properties of the RDBMS in use with the master domain manager - for example, to change the database name, server host, or port used to communicate with RDBMS
- ChangeHostProperties

  Used to change the workstation host name in the WebSphere Application Server configuration files, or to change the TCP/IP ports used by WebSphere Application Server.
- ChangeSecurityProperties

  Used to change the WebSphere Application Server security settings related to the use of Tivoli Workload Scheduler. In particular these settings are as follows:
  - General security settings, such as designating the user registry (local operating system or LDAP)
  - Local operating system user registry settings (user and password)
  - LDAP configuration settings
  - SSL key stores
  - DB2 user settings for Tivoli Workload Scheduler

Procedure for changing properties settings

Follow the procedure provided in this section when changing settings for any of the host, security, or data source properties. The steps in the procedure are as follows (see also Figure 4-8 on page 85):

1. Log on to the computer where Tivoli Workload Scheduler is installed as the following user:
   - UNIX: root
   - Microsoft Windows: Any user in the Administrator group
2. Access the directory *TWShome*/wastools.
3. Stop the WebSphere Application Server using the **`conman stopappserver`** command.
4. From the same directory, run the following script to create a file containing the current properties:
   - UNIX: **`show`**`<`*`property_type`*`>`**`Properties.sh >`** *`my_file_name`*
   - Microsoft Windows: **`show`**`<`*`property_type`*`>`**`Properties.bat >`** *`my_file_name`*

   where *<property_type>* is one of the following:
   - DataSource
   - Host
   - Security
5. Edit my_file_name with a text editor.

   Check the start of the file. The command might have written a message from the application server (WASX7357:) at the beginning of the file. Delete this message.
6. Change the value of the configuration parameters, according to requirements. You only modify only those parameters that should be changed.
7. Save the *my_file_name*.
8. Run the script:
   - UNIX: **`change`**`<`*`property_type`*`>`**`Properties.sh`** *`my_file_name`*
   - Microsoft Windows: **`change`**`<`*`property_type`*`>`**`Properties.bat`** *`my_file_name`*
9. Start the WebSphere Application Server using the **`conman startappserver`** command.
10. Check that the changes have been implemented in Tivoli Workload Scheduler.

To avoid the WebSphere Application Server being down for too long a time, you can postpone stopping WebSphere Application Server until after step 7 on page 87. The process of postponing is illustrated in Figure 4-9.

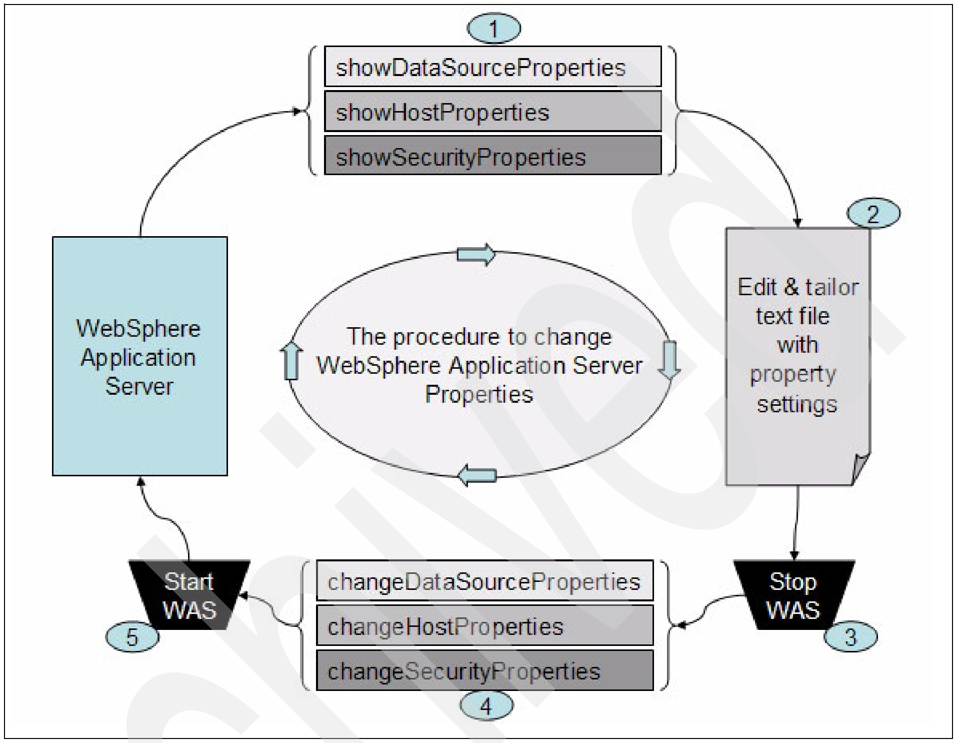

1
showDataSourceProperties
showHostProperties
showSecurityProperties
2
Edit & tailor text file with property settings
WebSphere Application Server
The procedure to change WebSphere Application Server Properties
Stop WAS
3
changeDataSourceProperties
changeHostProperties
changeSecurityProperties
4
Start WAS
5

Figure 4-9   WebSphere Application Server postponing process

Note: In Tivoli Workload Scheduler Version 8.4, you can stop and start the WebSphere Application Server with the new **`conman stopappserver`** and **`conman startappserver`** commands, respectively. It is also possible to stop and start WebSphere Application Server using the **`stopWas.sh`** (**`stopWas.bat`** on Microsoft Windows) and **`startWas.sh`** (**`startWas.bat`** on Windows) commands.

The following sections provides examples of how you can change the WebSphere Application Server properties using the procedure illustrated in Figure 4-9.

Change port number used for Job Scheduling Console

Suppose the network configuration department has decided that port 33106 specified during the installation of Tivoli Workload Scheduler and used by the Job Scheduling Console to access the UNIX master domain manager should be changed to 35117.

You can define the port for the Job Scheduling Console on the host properties settings. To change the port number, perform the following steps (see also Figure 4-9 on page 88):

1. Log on to the computer where Tivoli Workload Scheduler is installed as the UNIX root user.
2. Access the directory *TWShome*/wastools.
3. From the same directory, run the following script to create a file containing the current properties:

   `showHostProperties.sh >` *`my_file_name`*
4. Edit *my_file_name* with a text editor.
   - Check the start of the file. The command might have written a message from the application server (WASX7357:) at the beginning of the file. Delete this message.
   - Find the line with `bootPort=33106`. The bootPort is the port used by the Job Scheduling Console to connect to WebSphere Application Server and Tivoli Workload Scheduler.
5. Change the value of the bootPort configuration parameter to 35117.

   Example 4-10 shows how *my_file_name* may look after the change.

Example 4-10 my_file_name

```
################################################################
Host Configuration Panel
################################################################
oldHostname=zosaix3.lun.dk.ibm.com
newHostname=zosaix3.lun.dk.ibm.com

################################################################
Ports Configuration Panel
################################################################
bootPort=35117
bootHost=zosaix3.lun.dk.ibm.com
soapPort=33107
```

```
soapHost=zosaix3.lun.dk.ibm.com
httpPort=33104
httpHost=*
httpsPort=33105
httpsHost=*
adminPort=33112
adminHost=*
adminSecurePort=33113
adminSecureHost=*
sasPort=33108
sasHost=zosaix3
csiServerAuthPort=33109
csiServerAuthHost=zosaix3
csiMuthualAuthPort=33110
csiMuthualAuthHost=zosaix3
orbPort=33111
orbHost=zosaix3
```

6. Save *my_file_name*.
7. Stop the WebSphere Application Server using the **`conman stopappserver`** command.
8. Run the script:

   `changeHostProperties.sh` *`my_file_name`*
9. Start the WebSphere Application Server using the **`conman startappserver`** command.
10. Check that the changes have been implemented in the Tivoli Workload Scheduler. That is, check that it is possible to establish a connection from the Job Scheduling Console to Tivoli Workload Scheduler using the new bootPort number 35117.

Configure Job Scheduling Console to work with firewall

In this example, the TCP ports used by the WebSphere Application Server have to be customized to cross a firewall when Network Address Translation (NAT) is used.

The Job Scheduling Console and the WebSphere Application Server use the RMI/IIOP protocol to communicate. Some of the connections between the Job Scheduling Console and the WebSphere Application Server use JNDI and ORB services.

The bootPort, orbPort, and csiServerAuthPort must be opened on firewalls and forwarded to NATs:

- bootPort

  The port for the bootstrap or RMI. This port is used in the Job Scheduling Console to communicate with the Tivoli Workload Scheduler. The default value is 31117.

- orbPort

  The port used for RMI over IIOP communication. The default value is 31122. Set the orbPort to a controlled port when running in a NAT environment.

- csiServerAuthPort

  The server authentication port on which the Common Secure Interoperability Version 2 (CSIV2) service listens for inbound server authentication requests. The default value is 31120. Set the csiServerAuthPort to a controlled port when running in a NAT environment.

It is important that you correctly define the bootHost parameter. The bootHost parameter specifies the IP address or host name that is returned to the Job Scheduling Console and used to establish connections to the JNDI server. This address must be visible from the workstation where the Job Scheduling Console is running. This value is set during installation. Change the value to the public IP address or host name when you are running in a NAT.

Depending on the firewall definition and how host names are resolved in the network, you may have to define orbHost and csiSeverAuthHost.

To change the settings, follow the same procedure provided in the preceding example and illustrated in Figure 4-9 on page 88.

Change host name used in WebSphere Application Server

In this example, you change the host name zosaix3 used in the WebSphere Application Server on the UNIX master domain manager to the fully qualified host name zosaix3.lun.dk.ibm.com. One easy way to do so is by using the oldHostname and newHostName parameters on the host properties file.

To change the host name, follow these steps (see also Figure 4-9 on page 88):

1. Log on to the computer where Tivoli Workload Scheduler is installed as the UNIX root user.
2. Access the directory *TWShome*/wastools.
3. From the same directory, run the following script to create a file containing the current properties:

   ```
   showHostProperties.sh > my_file_name
   ```

4. Edit *my_file_name* with a text editor.
5. Check the start of the file. The command might have written a message from the application server (WASX7357:) at the beginning of the file. Delete this message and delete all lines except the following lines:

```
################################################################
Host Configuration Panel
################################################################
oldHostname=zosaix3
newHostname=zosaix3
```

6. Change the value of the `newHostname` configuration parameter to the following:

   `zosaix3.lun.dk.ibm.com`

   Here is how *my_file_name* looks after the change:

```
################################################################
Host Configuration Panel
################################################################
oldHostname=zosaix3
newHostname=zosaix3.lun.dk.ibm.com
```

7. Save the *my_file_name*.
8. Stop the WebSphere Application Server using the **conman stopappserver** command.
9. Run the script:

   changeHostProperties.sh *my_file_name*

10. Start the WebSphere Application Server using the **conman startappserver** command.

Check that the changes has been implemented in Tivoli Workload Scheduler. That is, check that it is possible to establish connection from the Job Scheduling Console to Tivoli Workload Scheduler using the new host name.

You can also check that the host name has been changed by running:

showHostProperties.sh > *my_file_check*

Verify that the host name has been changed in the *my_file_check* file (see Example 4-11).

Example 4-11 Output from showHostProperties.sh after changing host name

```
################################################################
Host Configuration Panel
################################################################
```

```
oldHostname=zosaix3.lun.dk.ibm.com
newHostname=zosaix3.lun.dk.ibm.com

################################################################
Ports Configuration Panel
################################################################
bootPort=35117
bootHost=zosaix3.lun.dk.ibm.com
soapPort=33107
soapHost=zosaix3.lun.dk.ibm.com
httpPort=33104
httpHost=*
httpsPort=33105
httpsHost=*
adminPort=33112
adminHost=*
adminSecurePort=33113
adminSecureHost=*
sasPort=33108
sasHost=zosaix3.lun.dk.ibm.com
csiServerAuthPort=33109
csiServerAuthHost=zosaix3.lun.dk.ibm.com
csiMuthualAuthPort=33110
csiMuthualAuthHost=zosaix3.lun.dk.ibm.com
orbPort=33111
orbHost=zosaix3.lun.dk.ibm.com
```

Comparing the output in Example 4-11 on page 92 with the host properties file in Example 4-10 on page 89, note that four instances of zosaix have been changed to zosaix3.lun.dk.ibm.com.

Modify WebSphere Application Server to use LDAP

This example provides the steps you can perform to change WebSphere Application Server to use an LDAP user registry instead of user definitions in the local operating system (this is the default after installing Tivoli Workload Scheduler) on a Microsoft Windows master domain manager.

To change the WebSphere Application Server to use LDAP, follow these steps (see also Example 4-11 on page 92):

1. Log on to the computer where Tivoli Workload Scheduler is installed as any Windows user in the Administrator group.
2. Access the *TWShome*/wastools directory.

3. From this directory, run the following script to create a file containing the current security properties:

   ```
   showSecurityProperties.bat > secProp.txt
   ```

4. Edit *secProp.txt* with a text editor.
5. Check the start of the file. The command might have written a message from the application server (WASX7357:) at the beginning of the file. Delete this message.
6. Change `activeUserRegistry` to `LDAP`.
7. Change the LDAP parameters in the LDAP section of the secProp.txt file.

   For further information about how to change and customize LDAP, refer to *IBM Tivoli Workload Scheduler Administration and Troubleshooting V8.4,* SC32-1275.

8. Save the secProp.txt file.
9. Stop the WebSphere Application Server using the **`conman stopappserver`** command.
10. Run the script:

    ```
    changeSecurityProperties.bat secProp.txt
    ```

11. Change the Windows services properties.

    On Windows, after WebSphere Application Server has been modified to use LDAP, it is important to change the properties of the Windows Services using the updateWasService.bat script - for example:

    ```
    updateWasService -userid tws83 -password zzzz –wasuser TEST_LDAP
    –waspassword xxxxxx
    ```

    where the userid and password options are the operating system user ID and password of the user running the WebSphere Application Server process; also where wasuser and waspassword refer to the logical user that is authorized to stop the WebSphere Application Server. The wasuser must be defined in the LDAP user registry.

12. Start the WebSphere Application Server using the **`conman startappserver`** command.
13. Check that the changes have been implemented in Tivoli Workload Scheduler. That is, check that it is possible to establish a connection from, for example, the Job Scheduling Console to Tivoli Workload Scheduler using a LDAP user.

4.6.2 Changing WebSphere Application Server trace properties

The Tivoli Workload Scheduler administrator uses the scripts *TWShome*/wastools/changeTraceProperties.sh (UNIX and Linux) and *TWShome*/wastools/changeTraceProperties.bat (Microsoft Windows) to adjust the application server trace settings. Run this script while the application server is running, and use it to change the trace level for both the Tivoli Workload Scheduler and the WebSphere Application Server code.

This script uses the properties file *TWShome*/wastools/TracingProps.properties, which defines the trace modes and enables traces of application server communications, either for the entire product or for the following specific features:

- Database
- Planner
- Command line
- Utilities
- Connector

To set the tracing level to include all the WebSphere Application server traces on a UNIX or Linux system, change the directory to the *TWShome*/wastools directory, and run the following command while the application server is running:

```
$ ./changeTraceProperties.sh -user username -password password -mode
tws_all
```

Note: This command should normally be run by root on UNIX and Linux or by a user that is member of the Administrator group on Microsoft Windows.

Example 4-12 shows the output produced by the changeTraceProperties.sh script.

Example 4-12 Setting tracing level to tws_all

```
WASX7209I: Connected to process "server1" on node DefaultNode using
SOAP connector;  The type of process is: UnManagedProcess
WASX7303I: The following options are passed to the scripting
environment and are available as arguments that are stored in the argv
variable: "[server1, DefaultNode, TracingProps.properties, tws_all]"
entering getTraceValue
properties file value for tws_all is  com.ibm.tws.*=all
Current trace specification is *=info
Current trace specification is now
*=info:com.ibm.tws.*=all
```

The trace file, trace.log, can be found in the directory:

```
TWShome/appserver/profiles/twsprofile/logs/server1
```

To reset the tracing to its default level, run changeTraceProperties.sh with -mode reset - for example:

```
$ ./changeTraceProperties.sh -user username -password password -mode
reset
```

The syntax for the changeTraceProperties.sh command is as follows:

- UNIX

  ```
  changeTraceProperties.sh -user TWSuser -password TWSuser_password
  -mode trace_mode
  ```

- Microsoft Windows

  ```
  changeTraceProperties.bat -user TWSuser -password TWSuser_password
  -mode trace_mode
  ```

Where the command attributes are as follows:

- -user *TWSuser*

  TWS installation user

- -password TWSuser_password

  Password defined for the TWS installation user

- -mode *trace_mode*

  Where *trace_mode* can be one of the following:

  – tws_all

    All Tivoli Workload Scheduler communications are traced.

  – tws_db

    All Tivoli Workload Scheduler database communications are traced.

  – tws_planner

    All Tivoli Workload Scheduler planner communications are traced.

  – tws_cli

    All Tivoli Workload Scheduler command-line communications are traced.

  – tws_utils

    All Tivoli Workload Scheduler utility communications are traced.

  – tws_conn

    All Tivoli Workload Scheduler connector communication are traced.

– tws_info

  No tracing - the default value.

4.6.3 Defining options for using the user interfaces

The final configuration step in relation to WebSphere Application Server is configuring the setup information required to connect to the WebSphere Application Server infrastructure. This information is necessary to use the Tivoli Workload Scheduler user interfaces (the command-line programs, the Job Scheduling Console, and the Tivoli Dynamic Workload Console) and to run the logman and JnextPlan scripts.

You must provide the following setup information to connect to the master domain manager using HTTP or HTTPS through the WebSphere Application Server infrastructure:

- Host name of the master domain manager.
- Port number user when establishing the connection with the master domain manager.
- Credentials, user name and password, of the TWSuser.
- Proxy host name used in the connection with the HTTP protocol.
- Proxy port number used in the connection with the HTTP protocol.
- Protocol used during the communication. This can be HTTP with basic authentication or HTTPS with certificate authentication.
- The timeout that indicates the maximum time the connecting user interface program can wait for the master domain manager response before considering the communication request to have failed.

All the preceding information, except the username and password, is stored in the *TWShome*/localopts properties file (for a description of the localopts file, see 4.4.2, "The local configuration file" on page 67).

After you install a Tivoli Workload Scheduler master domain manager, these options are defined in the localopts file for the MDM and are set so it is possible to connect to the master domain manager using WebSphere Application Server.

The useropts file

The useropts file contains the user credentials: username and password. This file is first read when the user connects to a Tivoli Workload Scheduler command-line program (composer or conman). In addition to containing the settings for the username and password, this file can optionally contain other settings that, if defined, overwrite those set in the localopts file. If the username and password is not specified in this file, you must specify them when invoking the command-line program.

Using the useropts file saves time because without a correct useropts file, the username and password must be specified every time the **composer** or **conman** commands are issued, for example:

```
composer -user username -password password display cpu=@
```

Because Tivoli Workload Scheduler supports multiple product instances installed on the same machine, more than one useropts file instance can exist. The possibility of having more than one useropts file, each with a different name, provides the ability to specify different sets of connection settings for users defined on more than one instance of the product installed on the same machine.

In the localopts file of each instance of the installed product, the useropts option identifies the file name of the useropts file that has to be accessed in the *user_home*/.TWS directory to connect to that installation instance.

If multiple Tivoli Workload Scheduler instances are installed on the same machine or server, make sure that different filenames are used for the files defined in the useropts option in the localopts files for the Tivoli Workload Scheduler instances.

Define the user name and password credentials in the useropts file

The user name and password credentials for the WebSphere Application Server connection can be provided in the following ways:

- Using the command line

  The user credentials can be provided using the -username and -password command-line parameters.

- Using the useropts file

  The username and password parameters can be specified in the useropts file - for example:

  ```
  # USERNAME and PASSWORD
  USERNAME = tws840
  PASSWORD = "ENCTRYPTMEPLEASE"
  ```

  The password must be enclosed in double quotes.

- At run time

  When the user credentials are not specified using the command-line parameters, or in the **useropts file**, the command-line requests them and automatically adds them to the useropts file.

Note: If the password is defined in the useropts file, it is encrypted the first time one of the Tivoli Workload Scheduler command-line programs (composer or conman) are called. After encryption the useropts file looks like this:

```
USERNAME = "tws840"
PASSWORD = "ENCRYPT:OtYlP8QlxTM="
```

If a new Tivoli Workload Scheduler user is defined and this user does not have a useropts file defined in *user_home*/.TWS, the user can create the useropts file manually or let Tivoli Workload Scheduler do it.

These are the steps for creating the useropts file for a new user without a useropts file in the directory /home/*user_name*/.TWS:

1. Enter `composer` (or `conman`) to start the Tivoli Workload Scheduler command-line interface. The following output from Tivoli Workload Scheduler is displayed:

   ```
   AWSBEH005E The user is not defined in the connection configuration
   file or the supplied command parameters.
   AWSBIA321I The credentials to connect to the remote server have not
   been specified.
   Specify your user name account:
   ```

2. Then enter the user account, `user_name`.

   Tivoli Workload Scheduler then prompts you for the password:

   ```
   Enter your password:
   ```

3. Enter the password.

   Tivoli Workload Scheduler prompts you for verification:

   ```
   Confirm your password:
   ```

4. Confirm the password.

   Tivoli Workload Scheduler then issues the following prompt:

   ```
   Do you want save the user name and password in your "useropts" file
   (enter "y" for yes, "n" for no)?
   ```

5. Respond to the prompt.

   If the answer is yes, Tivoli Workload Scheduler creates the useropts file in the /home/*user_name*/.TWS directory. This useropts file looks similar to the following:

   ```
   USERNAME = "dk010605"
   PASSWORD = "ENCRYPT:D+W1XJRomordmnbWRfwgcw=="
   ```

These two methods enable you to easily change the password for user_name:

- Manually edit the useropts file in the *user_name*/.TWS directory.

  Remember that the password is encrypted the first time either **composer** or **conman** is called.

- Delete the useropts file in the *user_name*/.TWS directory and then repeat the step 1 on page 99 through step 5 on page 99.

Note: If the useropts file exists with the correct user_name but wrong password, trying to access objects in Tivoli Workload Scheduler using either composer or conman results in an error message similar to the following:

```
AWSBEH021E The user "dk010605" is not authorized to access the
server on host "127.0.0.1" using port "33105"
```

5

Scheduling and operations

This chapter discusses the Tivoli Workload Scheduler behavior from the scheduling perspective. We discuss the following topics:

5.1 Working with the scheduling objects

This section explains how to work with the scheduling objects. We discuss the following topics:

- "Jobs" on page 102
- "Job streams" on page 106
- "Resources" on page 109
- "Prompts" on page 109
- "Users" on page 110
- "Working with composer" on page 111

5.1.1 Jobs

This section describes which properties and dependencies can be specified, when working with the job definition.

Job properties

This section describes which properties can be defined for jobs inside the job stream.

Job priority

This property is covered in detail in "Workstation fence" on page 138.

Requires Confirmation option

In 5.1.4, "Prompts" on page 109, we explain how to set up a *prompt dependency* on a job. A prompt dependency is similar to a "preceding breakpoint" for the job. It means that a job is not launched until you reply to a prompt. So prompts are used *before* the job launches.

The Requires Confirmation option is the opposite case. If a job has the Requires Confirmation option set, you must confirm the result state *after* the job has started. This option is used when you decide whether or not the job has run correctly. You may check the job's joblog to determine whether the job ended SUCCesfully or ABENDed.

Figure 5-1 demonstrates how you manually confirm the job status.

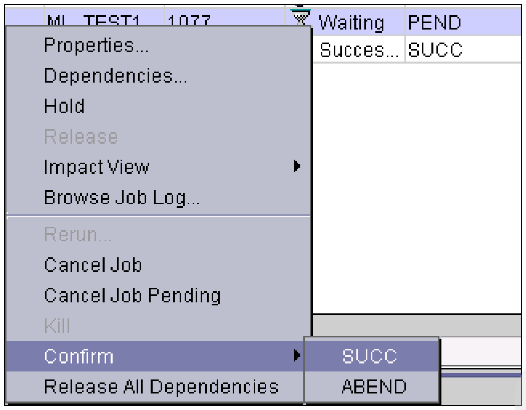
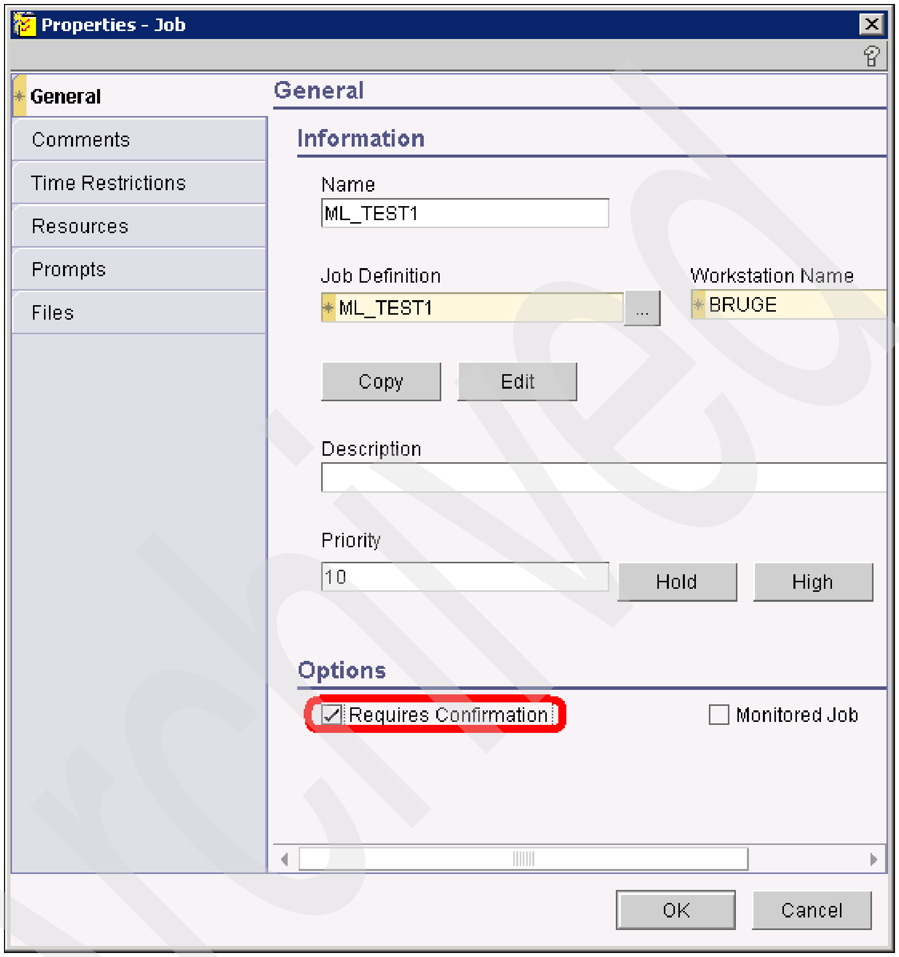


Figure 5-1 Confirming the final status

Note: The best use for the Requires Confirmation option is providing the capability of manually overriding (or confirming) the final status of a job (SUCC/ABEND) *after* the job has completed. Reply to prompts and Requires Confirmation can be performed by operators or by programs using the Tivoli Workload Scheduler conman command-line interface. The conman command line is often used to “automate” manual intervention.

The Requires Confirmation option can be set on jobs only. It cannot be set on job streams. Figure 5-2 shows where this option is located in the Job Scheduling Console.



Name: ML_TEST1

Job Definition: ML_TEST1

Workstation Name: BRUGE

Description:

Priority: 10

[✓] Requires Confirmation

[ ] Monitored Job

Figure 5-2 Requires confirmation

Job dependencies

This section describes which dependencies can be set on jobs inside the job stream. Furthermore, we discuss which dependencies can be set up on jobs placed into a job stream.

The following dependencies can be set on the jobs inside the job stream:

- Time dependency - This dependency determines time-related features, which are listed here:
  - Start time - Specifies the time when the job should start.

    This value is not specified by default, which means that the job starts when it is placed into the production plan (if the job has no other dependencies).
  - Repeat range - Specifies the time period when the job automatically resubmitted. The job is resubmitted into the same job stream.

    This value is not specified by default, which means that the job is submitted only once and is not repeated.
  - Latest start time - Specifies the latest time when the job is allowed to start. Using the latest start time, it is possible to define actions for the job, such as suppress, continue, and cancel. This value is not specified by default.
  - Termination deadline - Specifies the time when the job must complete. Jobs that have not yet started, or jobs that are still running, when the deadline time expires are considered late in the plan.

    This value is not specified by default.

    Note: The termination deadline marks the jobs as “late”; it *does not perform any other operations* on the job (such as canceling it, preventing it from starting, or adding dependencies of any sort).
- Prompt dependency - If a job has a prompt dependency, the job is not started until you positively (YES) confirm the prompt related to the job. See 5.1.4, “Prompts” on page 109 for details.
- File dependency - If a job has a file dependency, the job does not start until a specified file is created on the specified workstation. Note that the file checked against the existence of the job may be defined on workstation other than the one where the dependent job itself resides.
- Resource dependency - This dependency determines which logical resources the job consumes and what quantity of the resource the job consumes. Logical resources are used, for example, when multiple jobs are expected to share the same consumable resource of limited quantity.

  Note: Tivoli Workload Scheduler V8.4 uses only *local* resources; no global resources are available.

Note: Resources by themselves *do not manage the order* in which the jobs are executed. They do not allow the job to be launched if the current amount of available resources is *lower* than specified in the job's resource dependency.

The best way to manage the order in which the jobs are executed is to make one job dependent on another.

Recovery jobs

A *recovery job* can be defined when working with the job definition itself, not with the job properties inside the job stream. If a job has a recovery job specified, the recovery job is launched when the "original" job ends with an error.

Note: Typical use of a recovery job is running a script with corrective action or a script that performs notification of some sort, such as sending an e-mail or an SMS (Short Message Service).

5.1.2 Job streams

This section describes which properties and dependencies can be specified when working with the job stream definition.

Job stream properties

This section describes which properties can be defined for job streams.

Job stream priority

This property is covered in "Workstation fence" on page 138.

Job stream limit

The job stream limit specifies the number of jobs that can run inside the job stream at the same time. If a job stream limit is reached, the jobs that are about to be launched wait until some of the currently running jobs finish.The value of the job stream's limit can be modified in the production plan to meet current needs.

Note: The best use of the job stream limit is allowing only a specified number of jobs inside one job stream to run at the same time. You should use the job stream limit when you want to prevent many jobs from starting, and you cannot use dependencies for this task.

When the job stream limit is combined with job priority, you also can manage the order in which the jobs are launched. For instance, if 10 (independent) jobs are about to start, and the limit is set to 3, only the 3 jobs with the highest priority are launched. The other jobs wait until at least one of the current jobs finishes.

Draft option

Use the Draft check box to specify that the job stream cannot be submitted to the plan. The job stream is not added to the plan even if it has a valid run cycle for the current plan. In addition, it cannot be submitted manually.

Having the job stream in the "draft" state is one of the three major reasons why a job stream might not appear in the current production plan. See 5.2.4, "Determining job streams not added to the current plan" on page 126 for more information.

Carry forward option

In this section we explain how the workload is being carried forward to the next production plan. See 5.2.6, "Global options affecting production plan creation" on page 129 for details about the settings that influence carry forward behavior.

Some job streams may be still running (or have not yet started) when **JnextPlan** runs. Setting a job stream as *carry forward* means that the job stream with its jobs is transferred to the next production plan.

Note: Carried forward job streams are transferred to the new production plan. The names of the carried forward job streams remain unchanged.

Jobs that are running during the production plan extension are affected by the carry forward functionality as follows:

- Jobs from job streams that have been carried forward are placed in their original (carried forward) job streams.
- Jobs from job streams that have *not* been carried forward are placed in a special job stream called USERJOBS. See "USERJOBS job stream for lost jobs" on page 141 for more information.

Job stream dependencies

This section describes which dependencies can be set at the job stream level.

The following dependencies can be set on job streams:

- Time dependency - This dependency determines time-related features, such as the following:
  - Start time - Specifies the time when the job stream should start.

    This value is not specified by default, which means the job stream starts after it is placed into the production plan (if the job stream has no other dependencies).
  - Latest start time - Specifies the latest time at when the job stream is allowed to start. This value is not specified by default.
  - Termination deadline - Specifies the time when the job stream must complete. Job streams that have not yet started or that are still running when the deadline time expires are considered late in the plan.

    This value is not specified by default.

    Note: The termination deadline marks the job stream as "late"; it does not perform any other operations on the job stream (such as canceling a job stream, preventing it from starting, or adding dependencies of any sort).

- Prompt dependency - A job stream with a prompt dependency is not started, until you positively (YES) confirm the prompt related to the job stream. See 5.1.4, "Prompts" on page 109 for details.
- File dependency - A job stream with a file dependency does not start until a specified file is created on the specified workstation. Note that the file defined in the dependency checking can be on a workstation other than the one where the dependent job stream resides.

  The file dependency qualifiers are more specific on UNIX platforms.
- Resource dependency - This dependency determines which logical resources a job stream consumes and the quantity of the resource the job stream consumes. Logical resources are used when multiple job streams are expected to share the same consumable resource of limited quantity.

  Note: jTivoli Workload Scheduler V8.4 uses only *local* resources; no global resources are available.

Note: Resources by themselves do not manage the order in which jjob streams are executed. They do not allow the job stream to be launched if the current amount of available resources is lower than the amount specified in the job stream's resource dependency.

If you want to manage the order in which job streams executed, you must make one job stream dependent on another.

5.1.3 Resources

Resources are used for *semaphoring* (that is, signaling with flags) among jobs or job streams that consume common resources during their run.

Tivoli Workload Scheduler resource objects represent any logical consumable objects (either in the real or virtual world), such as backup tape drivers, printers, and licenses.

If you have multiple jobs or job streams that share a consumable resource with defined availability, you must perform following steps:

1. Define the *r*esource object in the Tivoli Workload Scheduler database. You must provide the name of the resource and the total quantity available.
2. Specify a resource dependency for the object that consumes the resource. The consuming object can be either a job or a job stream.
3. Define how many units of the resource the job or job stream should allocate.

Note: The best use for resource objects is to jshare common resources effectively.

5.1.4 Prompts

A prompt is a "preceding breakpoint" either to the job or to a job stream. The job or job stream with a prompt dependency cannot start until you confirm the prompt. See "Job dependencies" on page 104 and "Job stream dependencies" on page 108 for more details.

The two types of prompt are described here:

- Predefined prompts - These prompts are independent scheduling objects. They are defined in the jTivoli Workload Scheduler database as the prompt object types. They can be shared among multiple job streams.

- Ad-hoc prompts - These prompts are defined inside the particular job stream. They are stored in the Tivoli Workload Scheduler database inside the job stream definition. These prompts cannot be shared among multiple job streams.

Note: The best use of prompts is providing the capability of manually allowing (or disallowing) the job or job stream to launch. Without your positive response to the prompt, the dependent job or job stream does not start.

5.1.5 Users

This section describes the scheduling objects of the user type. These scheduling objects are necessary when you want to define jobs that will run on the Microsoft Windows platform.

Tivoli Workload Scheduler agents on Microsoft Windows platforms require the complete credentials for each user account used for launching a script or command. To run a Windows job under a particular user, this user must be already defined in the Tivoli Workload Scheduler database and propagated into plan.

The user definition must contain the following values:

- User name.
- Password.
- Workstation on which the user is defined. Wildcards are allowed; for example, an asterisk (*) matches all workstations.
- User's domain (optional).

The user must also be included in the production plan; otherwise the job fails.

Scheduling objects of type user are necessary only for jobs running on Microsoft Windows platforms. You need not define these objects for jobs running on UNIX platforms.

Note: The user object defines the user that runs a job on Microsoft Windows workstations. It has nothing to do with Tivoli Workstation Scheduler operators or administrators, which are used to access Tivoli Workstation Scheduler functions or objects. For example, you cannot log on the Tivoli Workstation Scheduler using a user object. You do so with a Tivoli Workstation Scheduler Admin or Operator user ID.

5.1.6 Working with composer

This section describes how to interact with the Tivoli Workload Scheduler database using the command-line interface. We focus on the **composer** command.

Using the `composer` command, you can list, view, add, modify, and delete scheduling objects.

Note: Changes made to the database are propagated into the production plan if either of these conditions occurs:

- FINAL job stream has run.
- JnextPlan has been executed manually.

Changes are propagated to the production plan after either of these conditions occurs.

Access to the scheduling objects is driven by the access policy specified in the security file.

The following list includes the most important composer commands. We do not provide the full syntax including the possible command arguments; we focus on the first keyword following the `composer` command.

- `list` - Lists the objects of selected type.
- `create` - Creates a text file containing the objects of selected type.
- `extract` - Same as the `create` command.
- `display` - Provides functionality similar to that provided by the `composer create` or `composer extract` commands. Instead of creating a text file containing the object definitions, this command displays the definitions on the standard output.
- `lock` - Locks the object in the database, so that other users cannot modify it. Locked objects are not visible to other users when using the `composer create` or `vcomposer extract` commands. Users working with the Job Scheduling Console can see the locked objects but cannot modify them until the objects are unlocked.
- `uvnlock` - Unlocks objects that have been previously locked.

  Note: Locking and unlocking objects (either using the composer or Job Scheduling Console) is the only supported way that enables only one user at a time to modify the affected objects.

- **delete** - Deletes objects from the database.
- **modify** - Automates steps that you otherwise perform manually when modifying objects using **composer**. The **modify** command performs the following steps:
  a. Extracts the object definition to a temporary file
  b. Locks the object definition
  c. Opens an editor enabling you to modify the definition
  d. Replaces the definition in the database after you close the text editor
  e. Unlocks the object definition
- **new** - Enables you to add a new object definition of the desired type. It opens a text editor that already contains a template for the specified object type. You can modify the template. After you close the editor, the object is added to the database.
- **add** - Adds a new object definition from a text file.

  Objects with names that are already present in the database are considered modifications, and a prompt is issued. If you confirm the prompt, the object definitions are replaced.

  You can suspend prompting by specifying the `;noask` directive.

  Objects with names that are not yet present in the database are considered new and are added without prompting you.
- **replace** - Replaces the object definitions from a text file.

  Objects with names already present in the database are replaced without prompting you. Objects with names that are not yet present in the database are added without prompting.

Note: Commands **composer add** *file*`;noask` and **composer replace** *file* are equivalent. They both perform the following actions:
- Objects that are new are added.
- Objects that already exist in the database and are present in the text file are replaced.
- Objects that are in the database and are not present in the text file remain unchanged.

5.2 Scheduling the workload

This section explains how the workload is scheduled. We discuss the following topics:

- "Tivoli Workload Scheduler production plan" on page 113
- "Creating the production plan" on page 114
- "Working with run cycles" on page 124
- "Determining job streams not added to the current plan" on page 126
- "Working with time zones" on page 126

5.2.1 Tivoli Workload Scheduler production plan

The Tivoli Workload Scheduler production plan is a set of information that describes the workload for the current production period. The production plan is generated on the master domain manager and then distributed to workstations in the Tivoli Workload Scheduler network. The production plan is stored in the Symphony file.

The production plan consists of the following:

- All jobs and job streams that should run within the current production period. Job streams are added when they have a valid run cycle for the current production period. See 5.2.4, "Determining job streams not added to the current plan" on page 126 to learn which job streams are not added to the current production plan.
- All dependencies for each job or job stream.
- All other scheduling objects that are related to the current production period (such as workstations and prompts).

The Symphony file is not a static file. It reflects the current events in the Tivoli Workload Scheduler network and thus changes dynamically.

When working with the production plan, we speak about two possible actions:

- You create the production plan in the following cases:
  - The production plan does not exist yet, and you are creating it for the first time.
  - You had to scratch the current production plan for some reason (for instance, because of a file system corruption). In this case, you must create the production plan again.

- You extend the production plan in following case:

  The production plan exists and is valid. You are either adding a consecutive production period or propagating important information from the Tivoli Workload Scheduler database (such as a new workstation definition that must be activated immediately).

Important: In the following sections, when we refer to production plan *creation*, we mean *extension* as well.

5.2.2 Creating the production plan

This section explains how the production plan is created or extended.

You can create the production plan only on the master domain manager or on the backup master domain manager that is currently acting as a master. You can create the production plan using two possible methods:

- Automatically during the execution of the FINAL job stream
- Manually by issuing the **JnextPlan** command from the command-line interface

We describe both ways in the following sections.

Creating the production plan using the FINAL job stream

This section explains how you can create the production plan automatically during the execution of the FINAL job stream. By default the FINAL job stream runs daily at 05:59 a.m. and extends the production plan by a 24-hour period.

The FINAL job stream consists of five jobs as shown in Figure 5-3.

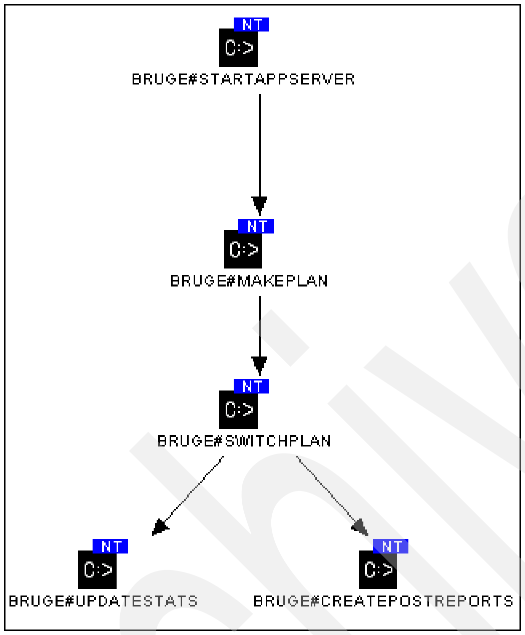


Figure 5-3 FINAL job stream

Each job in the FINAL job stream is responsible for a particular task, as we describe in the sections that follow.

Note: In the following sections, we describe the scripts referenced by the jobs in the FINAL job stream. We use the names of UNIX scripts. You must add the .cmd extension if you plan to use Microsoft Windows scripts.

STARTAPPSERVER

This job attempts to start the WebSphere Application Server if it is not already running.The following script is executed by this job:

```
<TWS_HOME>/wastools/startWas
```

The Tivoli Workload Scheduler startWas utility is launched from this script. This script invokes the WebSphere Application Server start method.

MAKEPLAN

This job creates a new plan or extends the preproduction plan. Then it creates the preproduction reports. The following script is executed by this job:

```
MakePlan
```

The following Tivoli Workload Scheduler utilities are launched from this script:

- `planman` - Based on the information in the database, this script creates a *preproduction plan* (also called an *intermediate plan*). The preproduction plan is stored in the Symnew file. The preproduction plan contains information about scheduling objects and their dependencies.
- `reptr -pre Symnew` - Creates the following reports:
  - Report 09A (preproduction summary generated from the preproduction plan)
  - Report 09B (preproduction detail generated from the preproduction plan)

SWITCHPLAN

This job merges previous Symphony file and the new Symnew file. It adds carry-forward job streams to the preproduction plan and thus creates the production plan.

The following script is executed by this job:

```
SwitchPlan
```

The following Tivoli Workload Scheduler utilities are launched from this script:

- `conman "stop @!@;wait;noask"` - Stops all agents in the Tivoli Workload Scheduler network.
- `stageman -log=schedlog/`*`<name_of_archived_plan>`* - Merges the previous plan with the preproduction plan created by planman (merges Symphony and Symnew). The new production plan is stored in Symphony and Sinfonia (the compressed version of Symphony).
- `planman confirm` - Confirms to the planner the switching of the production plan.
- `conman "stop @!@;wait;noask"` - Starts all agents in the Tivoli Workload Scheduler network. This mechanism starts distribution of the new Symphony file across the Tivoli Workload Scheduler network.

Note: When the SWITCHPLAN job is completed, the creation (or extension) of the production plan is completed, and the plan is distributed to target workstations.

UPDATESTATS

This job performs the following tasks:

- Logs job statistics.
- Checks the policies and if necessary extends the preproduction plan.
- Updates the preproduction plan reporting the job stream instance states.

The following script is executed by this job:

```
UpdateStats
```

The following Tivoli Workload Scheduler utility is launched from this script:

`logman schedlog/`*`<name_of_archived_plan>`* - Updates the preproduction plan, job history, and statistics.

CREATEPOSTREPORTS

This job creates post production reports. The following script is executed by this job:

```
CreatePostReports
```

The following Tivoli Workload Scheduler utilities are launched from this script.

- `reptr -post schedlog/`*`<name_of_archived_plan>`* - Creates following reports:
  - Report 10A (production summary generated from the previous plan)
  - Report 10B (production detail generated from the previous plan)
- `rep8 schedlog/`*`<name_of_archived_plan>`* - Creates Report 08 (job histogram generated from the previous plan).

Creating the production plan using JnextPlan

Tivoli Workload Scheduler V8.4 provides a command-line interface for creating (and extending) the production plan. We describe the location, behavior, and use of the relevant scripts in this section.

Note: We use the names of UNIX scripts in this section. You must add the .cmd extension if you are using Microsoft Windows scripts.

The following script creates or extends the production plan:

```
<TWS_HOME>/JnextPlan
```

By default, the JnextPlan script performs the same tasks as the FINAL job stream.

JnextPlan runs the following tasks and scripts:

- `conman "startappserver; wait"`
- `MakePlan <`*`arguments`*`>`
- `SwitchPlan`
- `CreatePostReports`
- `UpdateStats`

The major differences between the FINAL job stream and JnextPlan script are the following:

- FINAL job stream, which runs automatically, runs daily at 05:59 a.m. by default. You can adjust the start time of the FINAL job stream by modifying the "start at" time dependency.
- By default, the JnextPlan script is not referenced by any Tivoli Workload Scheduler job and thus does not run automatically. JnextPlan is a service script, and you should run it manually.
- By default, the FINAL job stream extends the production plan by a 24-hour period.
- By default, JnextPlan extends the production plan by a 24-hour period. Unlike the FINAL job stream, the JnextPlan script can take arguments from the command line. These arguments are then passed to the MakePlan script, which invokes the **`planman`** command. Using these arguments, you can specify the time period for the new production plan.

The following examples demonstrate the use of various arguments passed to the JnextPlan script. We explain how they affect production plan creation or extension.

- `JnextPlan` (without any arguments)

  If you do not specify any arguments, the production plan is extended by a 24-hour period.
- `JnextPlan -for HHMM`

  If you specify the `for` keyword, the production plan is extended by the period provided in the command line.

  The meaning of `HHMM` is:

  – Replace `HH` with two digits representing hours.

  – Replace `MM` with two digits represent minutes.

The following examples demonstrate the different values supplied with the `for` argument:

– `JnextPlan -for 0030`

  Extends the production plan by 30 minutes.

– `JnextPlan -for 9600`

  Extends the production plan by 96 hours (4 days).

– `JnextPlan -for 0000`

  Extends the production plan by 0 hours and 0 minutes. This command is useful when you want to update the current production plan with the latest changes made to the Tivoli Workload Scheduler database (for instance, a new workstation definition has been added).

– `JnextPlan -from mm/dd/yy -to /mm/dd/yy`

  Creates a new production plan affecting the specified period.

> **Important:** If you specify the `-from` argument, you always request the creation of a new production plan. It is not possible to use this argument when the current production plan exists and you want to extend it.

The following examples demonstrate the use of `-from` and `-to` arguments:

– `JnextPlan -from 05/20/08 -to 05/24/08`
  - Creates a new production plan starting on 05/20/08 and ending on 05/24/08.
  - This command fails if the current production plan exists.

– `JnextPlan -to /mm/dd/yy`

  Extends the production plan so that it ends on the specified date.

The following example demonstrates the use of the `-to` argument:

```
JnextPlan -to 05/24/08
```

This example extends the current production plan so that it ends on 05/24/08.

> **Note:** The argument `-for` is mutually exclusive with `-from` and `-to` arguments. You cannot combine `-for` with `-from` and `-to` in one **`JnextPlan`** command.

Differences among JnextPlan, MakePlan, and SwitchPlan

This section briefly discusses the differences among three important scripts that are accessible from the command-line interface:

- MakePlan - This script creates the preproduction plan. It invokes the **`planman`** command, which creates the preproduction plan.

  Completing MakePlan does not create the production plan.

- SwitchPlan - This script initiates the following actions:
  - Stops all workstations in the Tivoli Workload Scheduler network.
  - Merges the previous production plan and the preproduction plan created in the previous step. This is done by invoking the **`stageman`** command.
  - Starts all workstations in the Tivoli Workload Scheduler network. The script starts the distribution of the newly created production plan.

  When the SwitchPlan script completes, the production plan is created and distributed.

- JnextPlan - This “superscript invokes all subordinate scripts. MakePlan and SwitchPlan are invoked from the JnextPlan script. When the JnextPlan script completes, the production plan is created and distributed.

Note: The production plan is created (extended) *and* distributed after the completion of the SwitchPlan or JnextPlan scripts.

The production plan is not created *and* distributed by the completion of the MakePlan script.

Relationship of start time of the FINAL job stream and startOfDay

In this section, we explain the meaning and relationship of the start time of the FINAL job stream and the `startOfDay` setting.

The start time of the FINAL job stream determines when the FINAL job stream executes. It is a common “start at” dependency similar to that for any other job stream. When the FINAL job stream executes, it creates a new production plan (including carry-forward job streams) and distributes the production plan to workstations in the Tivoli Workload Scheduler network.

Note: The start time of the FINAL job stream by itself does not influence the start and end of the new production period.

The global option `startOfDay` determines when the production period starts. This is the essential argument for the **`planman`** command, which creates the preproduction plan for:

- Time specified in the global option `startOfDay`
- Date when the `startOfDay` occurred for last time in the production plan

  For instance, if the `startOfDay` is set to 06:00 a.m. and the production plan ends on 04/22/2008, the new production plan starts at 06:00 a.m. on 04/22/2008.

  The **`planman`** command by default extends the plan by a 24-hour period.

Note: No links exist in Tivoli Workload Scheduler V8.4 between the global option `startOfDay` and `start at` of the FINAL job stream. Production plan creation (and extension) can be run at any time, and `startOfDay` indicates only the moment when the new production period starts.

In addition, if you swap the values of `startOfDay` and the FINAL job stream `start at` parameter so that the FINAL job stream runs one minute *after* the "`startOfDay`, you do not create a loop. The FINAL job stream is added to the currently created plan with the date corresponding to the following day.

Default values and how to change them

This section describes the default values of the `startOfDay` global option and the time dependency of "`start at` in the FINAL job stream. The following list provides the default values and describes how to change them:

- Start time of the FINAL job stream
  - Default value - 0559
  - Modification - `composer modify FINAL` or using the Job Scheduling Console
- Global option `startOfDay`
  - Default value - 0600
  - Modification - `optman chg sd=<`*`HHMM`*`>`

If you have too many job streams carried forward to the next production plan, it is reasonable to adjust the `startOfDay` and `start at` of the FINAL job stream. To prevent large numbers being carried forward, you must determine the time when the least possible number of job streams runs. This moment is your `startOfDay`.

Note: You should set the start time of the FINAL job stream to one minute *before* the `startOfDay`. We recommend that you follow this rule, but it is not mandatory.

FINAL job stream in previous and current versions

This section describes the major differences in the FINAL job stream in the following versions of Tivoli Workload Scheduler:

- Prior Tivoli Workload Scheduler V8.3 (that is, versions up to 8.2.1): In the sections that follow, we refer to these versions as “previous” versions.
- Since the release of Tivoli Workload Scheduler V8.3 (that is, versions 8.3 and 8.4): The sections that follow, we refer to these versions as the “current” versions.

FINAL job stream in previous versions

In the previous versions, the FINAL job stream contained only the one JNEXTDAY job. This job invoked the following Tivoli Workload Scheduler utilities:

- schedulr
- compiler
- reptr, rep8
- stageman
- logman

All these tasks are issued from only one job.

FINAL job stream in current versions

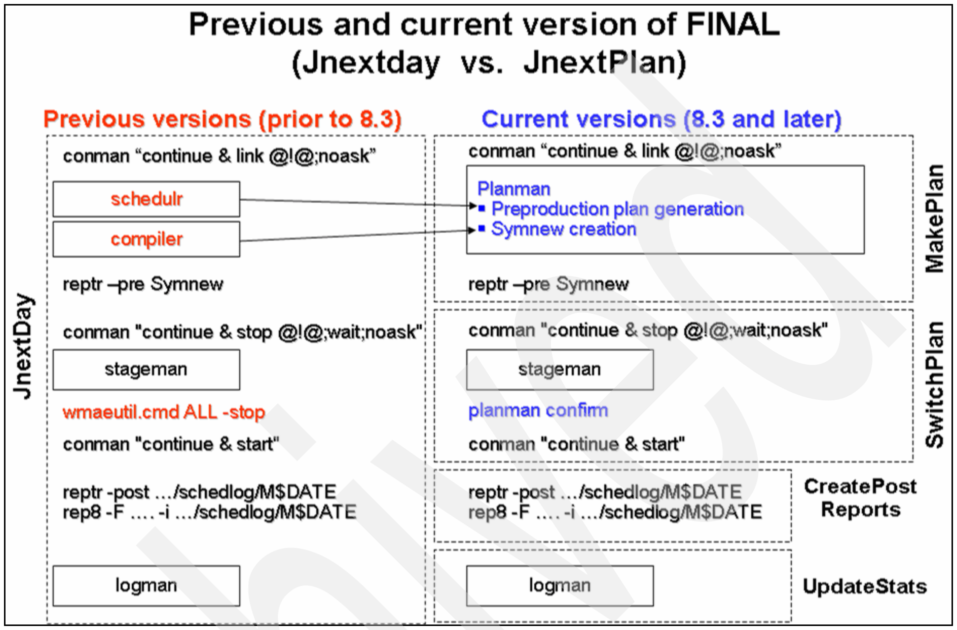
The FINAL job stream invokes the following Tivoli Workload Scheduler utilities:

- STARTAPPSERVER - startWas
- MAKEPLAN - planman, reptr
- SWITCHPLAN - stageman
- CREATEPOSTREPORTS - retpr, rep8
- UPDATESTATS - logman

Each task has a separate job.

Differences between the previous and current versions

Figure 5-4 shows the schema of both the previous and current versions of the FINAL job stream.



Previous and current version of FINAL (Jnextday vs. JnextPlan)
Previous versions (prior to 8.3)
Current versions (8.3 and later)
JnextDay
conman "continue & link @!@;noask"
schedulr
compiler
reptr –pre Symnew
conman "continue & stop @!@;wait;noask"
stageman
wmaeutil.cmd ALL -stop
conman "continue & start"
reptr -post …/schedlog/M$DATE
rep8 -F …. -i …/schedlog/M$DATE
logman
conman "continue & link @!@;noask"
Planman
Preproduction plan generation
Symnew creation
reptr –pre Symnew
MakePlan
conman "continue & stop @!@;wait;noask"
stageman
planman confirm
conman "continue & start"
SwitchPlan
reptr -post …/schedlog/M$DATE
rep8 -F …. -i …/schedlog/M$DATE
CreatePost Reports
logman
UpdateStats

Figure 5-4 FINAL job stream in previous and current versions

The following differences are shown in the figure:

- The previous versions contained only one job that performed all tasks related to production plan creation. The previous versions did not enable production plan extension, and no preproduction plans were created.
- The current version contains five jobs, each for a separate task. The production plan can be extended, and the preproduction plan is created during the FINAL job stream execution.
- The following utilities are no longer included in the current versions:
  - schedulr
  - compiler

Note: The schedulr and compiler utilities from the previous versions are replaced by the planman utility in the current versions.

5.2.3 Working with run cycles

In this section we describe how to use job stream run cycles to schedule the workload.

Run cycles determine whether the job stream is automatically added to the current plan during execution of the FINAL job stream or the JnextPlan script. You can choose from the wide selection of available run cycle types. We describe the run cycle types in "Run cycle types" on page 125.

You can define more than one run cycle for a job stream. It is also possible for job streams not to have any defined run cycle. We describe these special cases in following sections.

Job streams without any run cycle

Job streams without any run cycle are *not* automatically added to any production plan. This is equivalent to the `On Request` keyword in the versions prior to Tivoli Workload Scheduler V8.3.

If you want to run such job streams, you must submit them manually.

Job streams with multiple run cycles

As stated previously, you can define more than one run cycle for a job stream. For example, you can define one daily run cycle for workdays and one weekly run cycle for Wednesdays. In this example, Wednesday is the day matching two different run cycles. In the following sections, we describe the different behavior based on time dependencies.

The following information explains the behavior of the job stream for a day that matches multiple run cycles:

- If you do not specify any time dependencies in a run cycle, the job stream is added only once.
- If you do specify time dependencies for each run cycle and the time dependencies (start at) are equal, the job stream is added only once.
- If you specify time dependencies for each run cycle and the time dependencies (start at) are different, the job stream is added as many times as there are different dependencies.

  For instance, a job stream has two run cycles:

  – Daily run cycle on workdays at 13:15

  – Weekly run cycle on Mondays and Thursdays at 15:00

This job stream is added twice to production plans that affect Mondays and Thursdays because the run cycles have different time dependencies. The two added job streams have the same names (UNISON_SCHED) and different unique identifiers (UNISON_SCHED_ID).

R

Note: If the job stream has more than one concurrent run cycle for the current production plan, the job stream is still added into the production plan.

Run cycle types

This section lists the basic run cycle types. We also provide examples of run cycle usage.

The following list includes the basic run cycle types:

- Simple - Based on explicitly specified dates.

  Specify this run cycle when you want to run a job stream on specified dates that are not periodical.

- Calendar - Based on a previously defined calendar. This type is similar to the simple run cycle, but the calendar must be defined and stored in the database as an independent object.

  Specify this run cycle when you want to run a job stream on specified dates that are not periodical, and you have already defined a calendar that includes these dates (most probably because the calendar is used by multiple job streams).

- Daily - Specifies day-based frequency.

  Specify this run cycle when you want to run a job stream, for instance, once every three days. Another typical use of this run cycle is scheduling the job stream:

  – Every day
  – On workdays
  – On free days

- Weekly - Specifies week-based frequency

  Specify this run cycle when you want to run a job stream, for instance, each Monday.

- Monthly by date - Specifies monthly frequency based on date (that is, when the day occurs within the month)

  This run cycle is useful when you want to run a job stream, for instance, on the fifteenth day of every month.

- Monthly by day - Specifies monthly frequency based on day

  Specify this run cycle when you want to run a job stream, for instance, on the second Monday of every month.

- Yearly - Specifies year-based frequency

  Specify this run cycle when you want to run a job stream once a year.

5.2.4 Determining job streams not added to the current plan

Job streams are not added to the current plan for the following three reasons:

- The job stream is defined as Draft. Such a job stream cannot be submitted to the plan, neither automatically nor manually.
- The Valid from date has not occurred yet. For instance, you have specified in the job stream definition that the job stream is valid from 04/22/2008, but the current date is 10/18/2007.
- The run cycle of the job stream is not effective for the current production plan, or the job stream has no run cycle defined. For instance, you have specified a weekly run cycle for each Thursday, but the current production plan is effective for Monday and Tuesday.

Note: In the case when you have not defined a run cycle for a job stream, it is equivalent to the `On Request` keyword in the versions prior to Tivoli Workload Scheduler V8.3.

5.2.5 Working with time zones

This section explains the time zone feature. The time zone feature determines how Tivoli Workload Scheduler calculates the start time of particular jobs and job streams depending on the time zone of their workstations.

You can enable or disable the time zone feature by setting the global option `enTimeZone` either to YES or NO. By default the time zone feature is enabled, and thus Tivoli Workload Scheduler considers the time zone settings for each job, job stream, and workstation.

If you disable the time zone feature, Tivoli Workload Scheduler ignores time zone settings for any job, job stream, or workstation. However, you may experience incorrect time evaluation together with incorrectly displayed times in the conman interface or Job Scheduling Console.

The following sections describe Tivoli Workload Scheduler behavior when the time zone is enabled.

Time zones evaluation

This section describes how Tivoli Workload Scheduler V8.4 evaluates different settings related to the time zone feature. We also describe the best practices for working with time zones.

Supported scheduling objects

Time zones can be defined for the following scheduling objects:

- Job streams
- Jobs
- Workstations
  - Fault-tolerant agents (including master domain manager and backup master domain manager)
  - Standard agents
  - Extended agents

Default values determination

When you do not explicitly define the time zone for a scheduling object, it takes a default value based on following rules:

- The master domain manager (or backup master domain manager) without time zone specification inherits the time zone setting from the operating system where it is installed.
- A workstation without time zone specification inherits the time zone from the master domain manager.
- An extended agent without time zone specification inherits the time zone from the master domain manager.
- A job without time zone specification inherits the time zone from the workstation where the job is planned to run. If no time zone is defined for this workstation, the time zone is inherited from the master domain manager.
- A job stream without time zone specification inherits the time zone from the workstation where the job stream is planned to run. If no time zone is defined for this workstation, the time zone is inherited from the master domain manager.

Important: We recommend you set the time zone for each workstation of your Tivoli Workload Scheduler network. Make sure that the time zone you specify in the workstation definition is the same as the time zone set in the operating system. If not, you may experience unexpected inconsistencies in production processing.

Time zone naming convention

Tivoli Workload Scheduler supports two different naming conventions for time zone names:

- Short time zone names - three-character notation
- Long time zone names - area/city notation

Note: Short time zone names are supported in Tivoli Workload Scheduler V8.4, but their support may be removed in future versions. We recommend you use long time zone names.

Table 5-1 lists sample time zone names with their relationship to the GMT (UCT) time zone.

Table 5-1 Sample time zone names

Long name	Short name	Description	Relative to GMT
Europe/Paris	ECT	European Central Time	GMT+1:00
Europe/Istanbul	EET	European Eastern Time	GMT+2:00
America/New_York	EST	Eastern Standard Time	GMT-5:00
America/Detroit	EST	Eastern Standard Time	GMT-5:00
America/Chicago	CST	Central Standard Time	GMT-6:00
Asia/Calcutta	IST	India Standard Time	GMT+5:30

The complete list of supported long time zone names is included in the *IBM Tivoli Workload Scheduler Reference Guide V8.4*, SC32-1274.

Note: The long naming convention gives you more opportunity to determine the correct time zone. As you can see in Table 5-1 on page 128, multiple long names can match one time zone (for instance, America/New_York and America/Detroit both refer to Central Standard Time).

However, even if the list of possible names is long, not all major cities are included. For instance, for workstations located in Washington, D.C., you must specify the time zone name America/New_York.

Time zone example

This section demonstrates how the time is evaluated if an extended agent points to a target system located in a different time zone.

Suppose you have the following environment:

- Master domain manager in Prague (GMT+1).
- The extended agent points to the PeopleSoft system in Istanbul (GMT+2). This extended agent has no time zone set.
- The PeopleSoft job is defined to run on the extended agent at 9:30 a.m. No explicit time zone setting is passed to the job.

The following evaluation is performed inside Tivoli Workload Scheduler:

- The extended agent has no time zone defined and therefore inherits the time zone from the master domain manager. Thus, the time zone of the extended agent is GMT+1.
- A job specified to run on the extended agent has no time zone defined. Thus, it inherits the time zone from the workstation where the job should run (from the extended agent and therefore from the master domain manager).

If the job starts at 9:30 a.m. on the extended agent, the following occurs:

- The time difference between the extended agent (GMT+1) and the target PeopleSoft system (GMT+2) is one hour.
- From the extended agent's perspective, the job has started on the PeopleSoft system at 9:30 a.m. (relative to GMT+1 - European Central Time).
- From the perspective of the target PeopleSoft system, the job has started at 10:30 a.m. (relative to GMT+2 - European Eastern Time).

No additional shift occurs between the master domain manager and the extended agent because the time zone has been inherited. The only time difference is between the extended agent and the target PeopleSoft system. Thus, when looking at this question from local perspectives:

- Tivoli Workload Scheduler launches the job at 9:30 a.m., the local workstation's time.
- PeopleSoft launches the job at 10:30 in its local time zone.

We describe the global options that influence how Tivoli Workload Scheduler handles time zones in "Time zones related to global options" on page 131.

5.2.6 Global options affecting production plan creation

This section describes which global options impact the creation of the production plan.

Note: No global option name starts with the "set" prefix.

planman- and logman-related global options

The following list describes the global options affecting the behavior of **planman** and **logman** during the creation of the production plan:

- startOfDay (sd) - Determines the start of the production period. See "Relationship of start time of the FINAL job stream and startOfDay" on page 120 for details.

  The default value is 0600 (06:00 a.m.).

- logmanMinMaxPolicy (lm) - Determines which policy is considered when evaluating the minimum and maximum duration of the job.

  Possible values:

  – elapsedTime - The time between the start and the end of the job.

  – cpuTime - The time when the job has been running (that is, when the operating system assigned CPU time to that job).

  – Both (default) - Both elapsed time and CPU time are separately logged.

Carry forward-related global options

The following list describes the global options related to the carry forward feature:

- enCarryForward (cf) - Determines whether or not the job stream is carried forward.

  Possible values:

  – all (default) - All job streams are carried forward during JnextPlan, regardless, whether or not the option `Carry Forward` has been set.

  – yes - Only a job stream with the `Carry Forward` flag is carried forward during Jnextplan.

  – no - No job streams are carried forward during JnextPlan regardless of whether or not the option `Carry Forward` has been set.

- carryStates (cs) - Can narrow the jobs that should be carried forward only to jobs in defined states (for instance only jobs in "abend,exec,hold,intro" states).

  The default value is null (all states).

- enCFInterNetworkDeps (ci) - Determines whether the internetwork dependencies are carried forward.

  Possible values:

  – yes (default) - All external job stream dependencies are carried forward.

  – no - The external job stream dependencies are not carried forward.

- enCFResourceQuantity (rq) - Determines whether the resource dependencies are carried forward.

  Possible values:

  – yes (default) - If a resource is in use at the moment of the merging of the previous and new production plans, the amount of consumed resources is carried forward to the new production plan.

  – no - The number of consumed resources is not carried forward to the new production plan.

Time zones related to global options

The following list describes the global options related to the time zone feature:

- enTimeZone (tz) - Determines whether time zone management is enabled.

  Possible values:

  – yes (default) - Time zone management is enabled. Values assigned to the time zone settings (in workstation, job, and job stream definitions) are used to calculate the time when the jobs and jobs streams are run on the target workstations.

  – no - All time zone settings (in workstation, job, and job stream definitions) are ignored.

- enLegacyStartOfDayEvaluation (le) - Determines how the global option `startOfDay` is evaluated across the Tivoli Workload Scheduler network.

  Possible values:

  – no (default) - The value assigned to the `startOfDay` global option on the master domain manager is not converted to the local time zone set on each workstation across the network.

  – yes - The value assigned to the `startOfDay` global option on the master domain manager is converted to the local time zone set on each workstation across the network.

5.3 Working with the production plan

This section describes how to work with the current production plan. We discuss the following topics:

- “Working with conman” on page 132
- “Working with planman” on page 135
- “Canceling jobs and the impact on successors” on page 135

- “Getting similar output from conman and Job Scheduling Console” on page 136
- “Default values and how to change them” on page 121
- “Switching to an alternate plan” on page 137
- “Managing the workload on workstations” on page 138
- “Distinguishing multiple occurrences of job streams” on page 140
- “Special job streams in the production plan” on page 141

5.3.1 Working with conman

This section describes how to interact with the production plan using the command-line interface. We focus on the **`conman`** command.

conman examples

Table 5-2 displays important conman examples. We cover the following:

- Listing jobs (unfiltered)
- Listing jobs in particular states
- Canceling jobs
- Altering a job’s priority
- Releasing a job from particular dependency
- Changing the password of a Microsoft Windows user
- Changing a limit of the workstation

Table 5-2 conman examples

Purpose	Conman syntax
Viewing all jobs from all workstations	`conman “sj @#@”`
Viewing jobs in a specific internal state from all workstations	`conman “sj @#@+state=<state>”`
Canceling jobs with a specific name on all workstations in all job streams	`conman "cj @#@.<job_name>"`
Canceling jobs with a specific name and in specific internal states on all workstations in all job streams	`conman "cj @#@.<job_name>+state=<state1>,<state2>"`
Altering job priority	`conman "altpri <workstation>#<job.stream>.<job_name>; <priority>"`

Purpose	Conman syntax
Deleting dependency from a job	`conman "deldep <workstation>#<job.stream>.<job_name>;dependency"`

Be aware of the following points concerning job selectors:

- Wildcards for the **`conman`** command have following meaning:
  - @ - Any string (equivalent to the asterisk (*) in UNIX and Microsoft Widows terminal windows)
  - ? - Any character
- A fully qualified job name is *`workstation#job_stream.job`*.
- A fully qualified job name (including a list of internal states) is *`workstation#job_stream.job+state1,state2`*.
- The number character (#) separates the workstation from the job or job stream specification.
- A dot character (.) separates the job stream from the job.
- A plus sign (+) separates the job from its internal states.
- A comma (,) separates the desired internal states.

The job states in the Job Scheduling Console are different from internal states used by **`conman`**. Table 5-3 matches the most important Job Scheduling Console states and internal states.

Table 5-3 Job Scheduling Console internal states

Job Scheduling Console state	Internal state
Successful	SUCC
Error	ABEND
Error	FAIL

Note: Be aware that the Error status displayed in the Job Scheduling Console is represented by the two following internal states:

- ABEND
- FAIL

This is important when specifying the conman job selector that affects all Error jobs.

The following list provides several **conman** usage examples:

- Viewing jobs in Job Scheduling Console Successful state:

  ```
  conman "sj @#@+state=SUCC"
  ```

- Viewing jobs in internal ABEND state:

  ```
  conman "sj @#@+state=ABEND"
  ```

- Viewing jobs in Job Scheduling Console Error state:

  ```
  conman "sj @#@+state=ABEND,FAIL"
  ```

- Canceling all jobs starting on ACCOUNTING in the Job Scheduling Console Error state on all workstations in all job streams:

  ```
  conman "cj @#@.ACCOUNTING@+state=ABEND,FAIL"
  ```

- Setting the priority to 10 for all jobs with name REPORT on all workstations in all job streams:

  ```
  conman "altpri @#@.REPORT;10"
  ```

- Releasing a job with the name BACKUP from resource dependency on a resource with the name TAPE:

  ```
  conman "release job=@#@.BACKUP; needs=TAPE"
  ```

Note: When specifying the job selector for other purposes than displaying the job, you must provide a job stream name (at least as a wildcard).

- Changing the password of a Microsoft Windows user:

  ```
  conman "altpass <CPU>#<username>;password"
  ```

  This command changes the user's password in the current plan only! If you want to make a permanent change, you must update the password in the Tivoli Workload Scheduler database as well.

  Remember that scheduling objects of type user are necessary for the Microsoft Windows platform only.

Note: If password is not specified, **`conman`** prompts for a password and a confirmation.

- Changing a limit on all workstations in all domains to 0

  ```
  conman "lc @!@;0;noask"
  ```

5.3.2 Working with planman

This section describes capabilities of the planman utility. The following tasks can be accomplished by planman:

- Creating or extending the preproduction plan (also called the *intermediate* plan)
- Creating or extending the trial plan
- Creating or extending the forecast plan
- Resetting the production plan
- Resetting both the production plan and preproduction plan so that the production is created from scratch
- Unlocking database entries if they remained locked

Important: The planman command-line interface enables you to only unlock the Tivoli Workload Scheduler database. The locking process occurs automatically during preproduction plan creation.

The planman command-line interface gives you the capability of unlocking the Tivoli Workload Scheduler database if it remains locked, for instance, because of an error during preproduction plan creation.

- Deploying event rules (for an event-driven workload automation feature - new in Tivoli Workload Scheduler V8.4)

5.3.3 Canceling jobs and the impact on successors

In “conman examples” on page 132 we demonstrate how to use **`conman`** for canceling jobs. In this section we describe how a job cancellation influences the job’s successors.

The following statements are important for understanding what happens after job's cancellation:

- If any job is canceled, it releases its successors from their FOLLOWS dependency.
- The successor jobs start immediately when they meet all other dependencies.

In general, if a job's predecessor has been canceled, then the job is no longer dependent on its predecessor. Thus the following statements are *not* true:

- If a predecessor job is canceled, the successor gets canceled, as well.
- If a predecessor job is canceled, the successor can never run.

5.3.4 Getting similar output from conman and Job Scheduling Console

This section briefly describes how to create similar outputs in conman and in the Job Scheduling Console. Both conman and the Job Scheduling Console can display and filter scheduling objects based on various criteria.

Note: The Job Scheduling Console and conman are equivalent for most viewing purposes. Obtaining the desired output occurs through using conman or Job Scheduling Console.

The following sections include several examples describing how to receive similar outputs from conman and the Job Scheduling Console.

Displaying jobs in a particular status

If you want to display jobs in the internal state SUCC in both conman and in the Job Scheduling Console, perform the following actions:

- conman - Run the following command:

  ```
  conman sj "@#@+state=SUCC"
  ```

  The number sign (#) is the delimiter separating the workstation and the job.
- Job Scheduling Console - Create the following filter:
  - job stream=*
  - job=*
  - workstation=*
  - internal state=SUCC

Listing all workstations in all domains

If you want to display all workstations in all domains, perform the following:

- conman - Run the following command:

  ```
  conman "sc @!@"
  ```

  Note: The delimiter separating the domain and workstation is the exclamation point (!), not the number sign (#).

- Job Scheduling Console - This view is provided by default in Status of all workstations.

5.3.5 Jobs in ABEND and FAILED status

This section describes the special job status - FAILED. The status FAILED is different from the status ABEND. Both of them mean an error condition, and both are displayed in the Job Scheduling Console in the state ERROR.

The difference between the FAILED and ABEND state is as follows:

- The ABEND state is caused when a job has started the referenced command or script, and it has ended with a return code that does not represent the SUCC state.
- A FAILED state is caused by the following:
  - The script or command that is referenced by the job does not exist.
  - The script or command that is referenced by the job is not executable.
  - The user account that is used for running the job does not exist on the workstation.
  - The user account that is used for running the job does not have sufficient authorization to run the script or command referenced by the job.

Note: The batchman process on the target workstation performs no checks against these conditions prior to launching the job. The jobs are submitted, regardless of whether their users or scripts exist. Note that there will be no job number and no job output for FAILED jobs.

5.3.6 Switching to an alternate plan

Here is how you can switch to the previous production plan so you can view it in the Job Scheduling Console: go to the Action list pane and click **Set Alternate Plan**. Select the appropriate engine and then select the desired plan from the list.

You can view the content of the alternate plan when you refresh your current view or switch to another view.

Note: An alternate plan is displayed at the moment it finishes. You always view the final state of the plan, not its beginning or any sort of playback.

5.3.7 Managing the workload on workstations

This section describes how to control the workload submitted to a particular workstation.

Workstation fence

If the workstation is under a heavy workload (or you expect that this situation is about to occur), you can modify a workstation fence. The value of the fence determines which jobs and job streams can start on the workstation.

The workstation fence is tightly related to the priority defined for each job and job stream. Only jobs or job streams with higher priority than the workstation fence can start.

Note: Priority is a job or job stream property, and fence is a workstation property.

The value of the workstation fence can be modified only in the production plan. The value of the job or job stream priority is defined in the database, and then it is propagated to production plan when the production is extended or created. The value of the priority can be altered in the plan as well.

Table 5-4 lists the default values of "fence" and "priority" and describes when the values are used.

Table 5-4 Default values for fence and priority

Property	Default value	When is the default value is used
Job priority	10	While creating the job definition in the database. The current value from the database is propagated to the plan when the job is submitted.

Property	Default value	When is the default value is used
Job stream priority	10	While creating the job stream definition in the database. The current value from the database is propagated to the plan when the job stream is submitted.
Workstation fence	0	While the machine is put into the plan for the first time, which can occur in following cases: ► A new workstation definition is added. ► A plan has been created from scratch.

Tivoli Workload Scheduler uses following names for specific values of “fence” and “priority”:

- HIGH - Equal to a value of 100
- GO - Equal to a value of 101

The following list contains important terms concerning “fence” and “priority”:

- Jobs with priority *lower or equal* to fence do not start. Jobs must *always* have a priority higher than the fence to start.
- If the workstation fence is set to GO, no job can start. Even jobs with their priority set to “GO cannot start because their priority is not higher.
- Jobs with priority 0 are in the state HELD. They cannot start, regardless of the value of fence.

Workstation limit

In “Workstation fence” on page 138, we have explained how to use a workstation fence to control the workload on the workstation. Using the workstation *limit* is another way to accomplish this task.

The workstation limit defines the number of jobs that can run on the workstation at the same time. When a workstation limit is reached, the jobs that are about to be launched wait until some of the currently running jobs finish. You can modify the value of a workstation’s limit only in the production plan.

The workstation limit is set to 0 when the workstation is added to the plan, or when the plan is created from scratch. You must alter this value in the plan to enable jobs to run on this workstation.

Note: The best use of a workstation's limit is to prevent additional jobs from starting when a machine is under heavy workload.

If the workstation limit is set to 0, only jobs with priority HIGH (100) or GO (101) can start on that workstation. No other jobs can start on that workstation, regardless of whether they have been released from all dependencies.

5.3.8 Distinguishing multiple occurrences of job streams

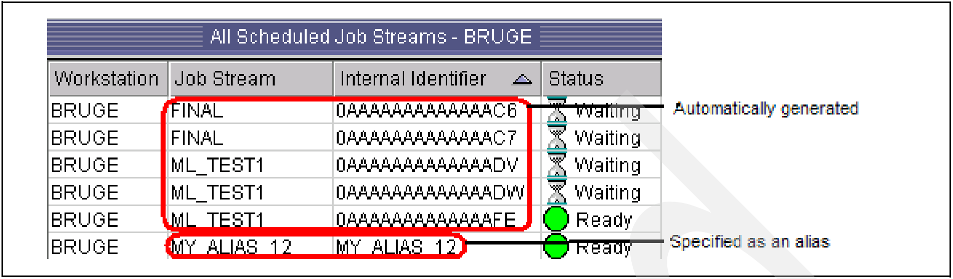
Since the release of Tivoli Workload Scheduler V8.3, it has been possible to submit multiple job streams with the same name into the same production plan. The representation of the job streams in the plan is as follows:

- UNISON_SCHED - The internal variable that represents the name of the job stream as it is defined in the database.
- UNISON_SCHED_ID - The unique identification of the job stream within the current production plan. The value of UNISON_SCHED_ID can be specified in two ways:
  - Automatically generated:
    - The job stream is added to the plan during the plan extension (JnextPlan).
    - The job stream is submitted without your specifying the job stream alias.
  - Manually specified: The job stream is submitted by you, and the job stream alias has been specified.

Note: As previously mentioned, multiple job stream instances with the same *name* can exist in the same plan at the same time because the unique key distinguishing job streams is the variable UNISON_SCHED_ID.

This statement also means that multiple job stream instances with the same *alias* cannot exist in the same plan - simply because the alias is stored in the UNISON_SCHED_ID variable.

Figure 5-5 shows the values of UNISON_SCHED_ID - either generated automatically or specified as the job stream alias.


All Scheduled Job Streams - BRUGE

Workstation	Job Stream	Internal Identifier	Status
BRUGE	FINAL	0AAAAAAAAAAAAAC6	Waiting
BRUGE	FINAL	0AAAAAAAAAAAAAC7	Waiting
BRUGE	ML_TEST1	0AAAAAAAAAAAAADV	Waiting
BRUGE	ML_TEST1	0AAAAAAAAAAAAADW	Waiting
BRUGE	ML_TEST1	0AAAAAAAAAAAAAFE	Ready
BRUGE	MY_ALIAS_12	MY_ALIAS_12	Ready

Figure 5-5 UNISON_SCHED_ID variable

5.3.9 Special job streams in the production plan

This section describes two special job streams in the production plan.

USERJOBS job stream for lost jobs

The job stream USERJOBS is used in special cases when Tivoli Workload Scheduler is aware of some running jobs, but it is no longer managing them. This state usually means that these jobs are not part of any current job stream but they are maintained by the jobman process.

The job stream USERJOBS is created in the following cases:

- Jobs running during JnextPlan were part of the job streams that have *not* been carried forward.
- The production plan has been reset and recreated while some of the jobs have been running.

In both cases, the jobs are “orphaned” because the current production plan does not include their “parent” job streams. Because of this situation, these jobs are put into the special job stream USERJOBS.

Note: Jobs from job streams that have been carried forward are placed in their job stream because these job streams are present also in the new production plan.

See “Carry forward option” on page 107 for more information.

JOBS job stream for ad-hoc submitted jobs

If an operator submits an ad-hoc job and does not explicitly specify the job stream name (submit into), the default job stream name for ad-hoc submitted jobs is JOBS.

5.4 Working with reports and statistics

This section describes how to work with traditional command-line interface reports. We also describe how to generate a forecast plan.

We cover the following topics:

- "Tivoli Workload Scheduler reports" on page 142
- "Displaying job states from a previous plan" on page 144
- "Creating a forecast plan" on page 147
- "Reports present in the FINAL job stream" on page 143

5.4.1 Tivoli Workload Scheduler reports

Table 5-5 lists Tivoli Workload Scheduler reports available through the command-line interface.

Table 5-5 Tivoli Workload Scheduler reports

Report name and purpose	CLI command
Report 01 - Job Details Listing	rep1
Report 02 - Prompt Listing	rep2
Report 03 - Calendar Listing	rep3
Report 04A - Parameter Listing	rep4a
Report 04B - Resource Listing	`rep4b`
Report 07 - Job History Listing	`rep7`
Report 08 - Job Histogram	`rep8`
Report 09A - Planned Production Summary	`reptr -pre -summary` *`<symphony_file_name>`*
Report 09B - Planned Production Detail	`reptr -pre -detail` *`<symphony_file_name>`*

Report name and purpose	CLI command
Report 10A - Actual Production Summary	`reptr -post -summary <symphony_file_name>`
Report 10B - Actual Production Detail	`reptr -post -detail <symphony_file_name>`
Report 11 - Planned Production Schedule	`rep11`
Report 12 - Cross Reference Report	`xref`

Note: Preproduction and post-production reports 09A-10B are accessible through the **`reptr`** command only. You must supply the correct switch (`pre` or `post`) to specify whether you want to generate a preproduction or actual report. If you do not supply either the `-summary` or `-detail` arguments, both summary and detailed reports are generated.

If you want to run a post-production report on an archived plan, you must provide also the plan file name.

An example of post-production report use is included in 5.4.3, “Displaying job states from a previous plan” on page 144.

5.4.2 Reports present in the FINAL job stream

This section describes the reports that are automatically launched during execution of the FINAL job stream.

The job stream FINAL consists of the following jobs:

- startappserver
- makeplan
- switchplan
- createpostreports
- updatestatistics

Two of these jobs launch the following reports:

- makeplan
  - Report 09A - Preproduction summary generated from the new plan
  - Report 09B - Preproduction detail generated from the new plan

- createpostreports
  - Report 10A - Actual production summary generated from the previous plan
  - Report 10B - Actual production detail generated from the previous plan
  - Report 08 - Job histogram generated from the previous plan

Note: Important preproduction and post-production reports can be displayed in the joblogs of the makeplan and createpostreports jobs.

Running preproduction and post-production reports manually

This section describes how to manually run the preproduction and post-production reports that automatically run within the job stream FINAL.

You can issue commands to run the following reports:

- Preproduction reports: `reptr -pre Symnew`
- Post-production reports: `reptr -post schedlog/<name_of_archived_plan>`

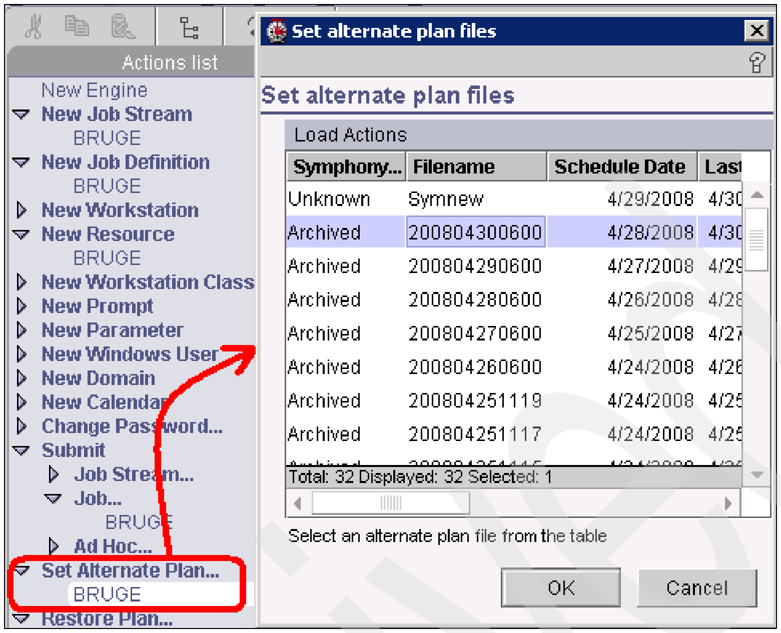
Note: Be aware that no reports run automatically *outside* the job stream FINAL. The only reports that run by default run inside the FINAL job stream.

5.4.3 Displaying job states from a previous plan

You can display jobs from a previous plan, together with their states (SUCC, ABEND, and so on), by using the following methods:

- Using the Job Scheduling Console:
  a. In the Action list pane, click **Set Alternate Plan**. Select the required engine and then select the desired plan from the list. The content of the alternate plan is displayed when you refresh your current view or switch to another view.
  b. Access the view that contains the desired job.
  c. View the job's status.

  Figure 5-6 on page 145 shows how to switch to alternate plan using the Job Scheduling Console.



Set alternate plan files

Symphony...	Filename	Schedule Date	Last
Unknown	Symnew	4/29/2008	4/30
Archived	200804300600	4/28/2008	4/30
Archived	200804290600	4/27/2008	4/29
Archived	200804280600	4/26/2008	4/28
Archived	200804270600	4/25/2008	4/27
Archived	200804260600	4/24/2008	4/26
Archived	200804251119	4/24/2008	4/25
Archived	200804251117	4/24/2008	4/25

Total: 32 Displayed: 32 Selected: 1

Select an alternate plan file from the table

Figure 5-6 Switching to alternate plan using Job Scheduling Console

- Using the **`conman`** command:
  a. Run **`conman`** without any parameters to enter the conman interface
  b. Use the **`listsym`** command to determine the number of the desired archived plan.
  c. Issue the **`setsym <archived_plan_number>`** command to switch to the alternate plan.
  d. Use the **`showjobs <jobselector>`** command to obtain the desired job state.

  Example 5-1 shows the process of following these steps.

Example 5-1 Getting job status from previous plan using conman

```
C:\Program Files\IBM\TWS\tws>conman
% listsym
Job Stream  Actual Start      Log      Run         Log
Date         Date   Time      Date     Num   Size   Num   Filename
04/29/08    04/29/08 06:00    04/30/08    31    45     1  M200804300600
Exp
04/28/08    04/28/08 06:00    04/29/08    30    36     2  M200804290600
Exp
```

```
04/27/08   04/27/08 06:00   04/28/08    29    36    3  M200804280600
Exp
04/26/08   04/26/08 06:00   04/27/08    28    35    4  M200804270600
Exp
04/25/08   04/25/08 11:19   04/26/08    27    36    5  M200804260600
Exp

% setsym 1
Scheduled for (Exp) 04/29/08 (#31) on BRUGE. Symphony file switched.
(1)sj @#@.sleep
(1)sj @#@.sleep

Workstation Job Stream       SchedTime  Job
BRUGE          #ML_TEST1          1502 04/25 ****   READY
                                   SLEEP  SUCC
```

- Using post-production reports
  a. Get the name of the archived Symphony file containing the desired production plan:

     Issue **`conman listsym`** to get the detailed list of archived plans.

     Be aware that you need to determine the archived Symphony *file name*, not the plan number.

  b. Run the following report and get the desired job state:

     `reptr –post -summary schedlog/<name_of_archived_plan>`

  Example 5-2 shows the process of following the preceding steps.

Example 5-2 Getting job status from previous plan using reptr

```
C:\Program Files\IBM\TWS\tws>conman listsym
Job Stream  Actual Start      Log      Run           Log
Date        Date   Time       Date     Num    Size   Num   Filename
04/29/08   04/29/08 06:00   04/30/08    31    45    1  M200804300600
Exp
04/28/08   04/28/08 06:00   04/29/08    30    36    2  M200804290600
Exp
04/27/08   04/27/08 06:00   04/28/08    29    36    3  M200804280600
Exp
04/26/08   04/26/08 06:00   04/27/08    28    35    4  M200804270600
Exp
04/25/08   04/25/08 11:19   04/26/08    27    36    5  M200804260600
Exp

C:\Program Files\IBM\TWS\tws>reptr -post -detail schedlog\M200804300600
```

```
Tivoli Workload Scheduler (Windows)/REPORTER 8.4 (20080317)
ITSO
Page    1
Report 10B   M200804300600                    Actual Production Detail
For 04/29/08
Job Name    Priority  Start Time       Run Time  Number  Status

Schedule   BRUGE#ML_TEST1 10     14:02(04/25/08)    0:03
READY
ML_TEST1       10     14:04(04/25/08)    0:01    #J1137   PEND
           SET            10     14:02(04/25/08)    0:04    #J1133
ABEND
           SLEEP          10     14:03(04/25/08)    0:02    #J1134
SUCC
```

5.5 Creating a forecast plan

In this section we describe how to generate the forecast plan.

You can generate the forecast plan using two methods:

- Using the Job Scheduling Console:
  a. In the Action list pane, click **Generate New Plan** and select the proper engine.
  b. In the pop-up window, select **Forecast** from the list of plan types.
  c. Specify the date and duration of the plan.

  Figure 5-7 on page 148 shows how to generate the forecast plan using the Job Scheduling Console.

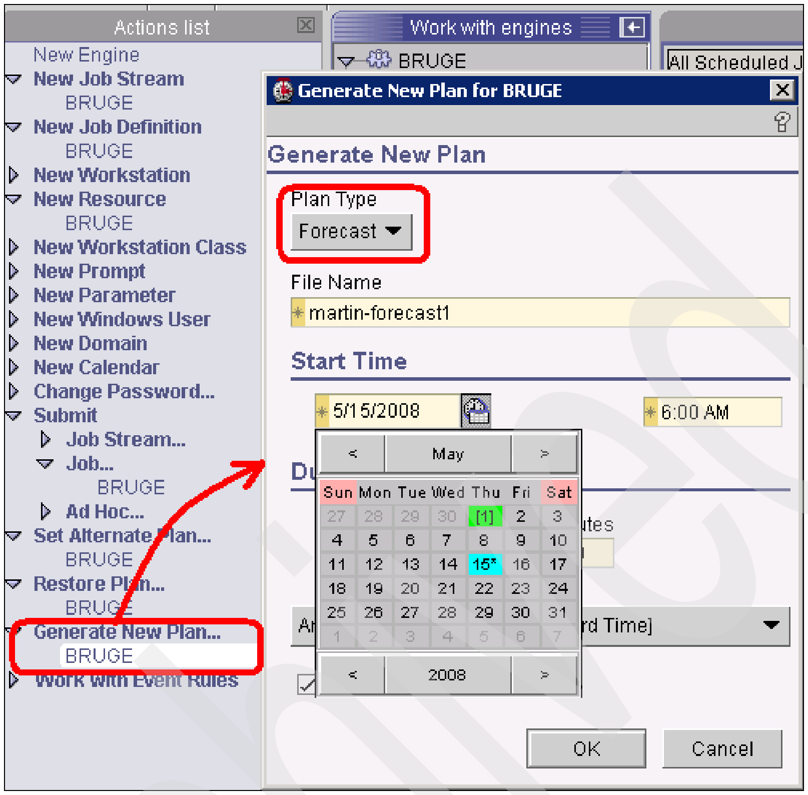


Figure 5-7   Generating forecast plan using Job Scheduling Console

- Using the **`planman`** command: **`planman crtfc`** *<date_and_range_specification>*

  It is possible to specify the range of the forecast plan in various ways, which are similar to arguments used in conjunction with the **`JnextPlan`** command. One way you can specify the range of a forecast plan is shown in Example 5-3.

Example 5-3   Creating a forecast plan using planman

```
C:\Program Files\IBM\TWS\tws>planman crtfc martin2-forecast -from
05/05/08 -to 05/06/08
Tivoli Workload Scheduler (Windows)/PLANMAN 8.4 (20080317) Licensed
Materials - Property of IBM(R)
5698-WSH
(C) Copyright IBM Corp 1998, 2007 All rights reserved.
US Government User Restricted Rights
Use, duplication or disclosure restricted by GSA ADP Schedule Contract
with IBM Corp.
IBM is a registered trademark of International Business Machines
```

```
Corporation in the United States, other countries, or both.
Installed for user "BRUGE\tws".
Locale LANG set to the following: "en"
AWSJPL710I During the creation of a forecast plan a temporary
preproduction plan is created automatically.
AWSJPL711I During the creation of a forecast plan, the planner has
successfully created a new temporary preproduction plan.

AWSJPL717I After a forecast plan has been created, the preproduction
plan that was used to build it has been deleted.
AWSJCL061I The forecast plan "martin2-forecast", renamed by planner as
"Fmartin2-forecast", has been successfully created.
```

5.6 Integrating with monitoring solutions

This section explains how to monitor the current production. We describe integration with traditional Tivoli products such as Tivoli NetView® and Tivoli Enterprise Console.

5.6.1 Source of monitoring information

The traditional way of integrating with monitoring products is accomplished by sending batchman events so that they can be received by the monitoring target. You can configure any fault-tolerant agent to send its batchman events to the monitoring product (such as Tivoli NetView and Tivoli Enterprise Console).

By default, no fault-tolerant agent (including the master domain manager) sends the batchman events. Thus, no integration is active by default.

5.6.2 Configuration file

You can configure the sending of batchman events in the file BmEvents.conf. The template for this file is stored in the following path:

- UNIX and Linux: <*TWS_HOME*>/OV/BmEvents.conf
- Microsoft Windows: <*TWS_HOME*>/config/BmEvents.conf

This template must be modified and copied into the directory path <*TWS_HOME*>/BmEvents.conf.

The template is located on each fault-tolerant agent, including the master domain manager. After the template is customized based on the environment, it should be copied into <*TWS_HOME*>.

Note: Any changes made to the BmEvents.conf file become active *after* restarting the affected Tivoli Workload Scheduler engine.

5.6.3 Integration types

This section describes the following integration types supported in a traditional integration:

- Text file - batchman events are written to a specified text file. The file is then analyzed by a log parser.

  This is the way the Tivoli Enterprise Console integration is accomplished. The batchman events are logged into a text file, and the Tivoli Enterprise Console logfile adapter parses this file.

  To force batchman to log to the text file parsed by the Tivoli Enterprise Console logfile adapter, the following must appear uncommented in the BmEvents.conf file:

  ```
  FILE=<path_to_file>
  ```

- Piping to the FIFO file - Batchman events are piped to the FIFO file, which is further processed by the background agent.

  Tivoli NetView integration is accomplished this way. The FIFO file is then processed by the magent process. Magent then forwards the batchman events as SNMP traps to Tivoli NetView.

  To force batchman to pipe to a FIFO file used by magent, the following must appear uncommented in the BmEvents.conf file:

  ```
  PIPE=MAGENT.P
  ```

5.6.4 Determining the integration sources

As previously described, any fault-tolerant agent can be integrated with a monitoring product. By default the master domain manager is informed about events from all workstations. Fault-tolerant agents are informed about events from all workstations only if they are set to full status.

Turning on integration on the master domain manager

Sending batchman events from the master domain manager is the preferred method because by default it is informed about events from all workstations.

To turn on integration that sends events from all Tivoli Workload Scheduler workstations, you modify BmEvents.conf. Turn on logging to a text file or piping to a FIFO file by inserting an uncommented line containing FILE or PIPE variables (see 5.6.3, “Integration types” on page 150 for details).

> **Note:** You need not uncomment the line containing the OPTIONS=MASTER variable because this value is the default on the master domain manager. When integration is switched on, the master domain manager sends information from all workstations by default.

Turning on integration on the fault-tolerant agent

In some cases. it is better to send batchman events from fault-tolerant agents as well. To configure the fault-tolerant agent to send information from *all workstations*, you must modify the workstation’s BmEvents.conf in the following way:

1. Turn on logging to a text file or piping to a FIFO file by inserting an uncommented line containing FILE or PIPE variables (see 5.6.3, “Integration types” on page 150 for details).
2. Uncomment the line containing the following text:

   ```
   OPTIONS=MASTER
   ```
3. Make sure that the workstation definition in the Tivoli Workload Scheduler database is set to Full Status.

5.6.5 Filtering the information only to key jobs

The amount of information sent to monitoring products can be huge in large Tivoli Workload Scheduler environments. To reduce this amount, perform the following steps:

1. Determine the important jobs and job streams.
2. Set them as the key jobs. In the Job Scheduling Console, the Monitored job flag or the Monitored job stream flag must be set.
3. Set the following value in BmEvents.conf:

   ```
   LOGGING=KEY
   ```

> **Note:** The default value is LOGGING=ALL, which means that if you do not change this value, information about all jobs and job streams is sent.

6

Administration and maintenance

In this chapter we describe how to perform regular administrative and maintenance tasks. We cover the following topics:

6.1 Sourcing the command-line interface environment

This section describes how to source the necessary command-line interface environment for most of Tivoli Workload Scheduler commands.

Setting up general environment variables

This section describes built-in scripts that can help you set the environment variables necessary for the Tivoli Workload Scheduler command-line interface.

Issue the following commands from the *<TWS_HOME>* directory:

- Microsoft Windows platform - **`tws_env.cmd`**
- UNIX and Linux platforms - **`.tws_env.sh`**

> **Important:** Note the use of a preceding dot (.) character for UNIX and Linux platforms.

These scripts set the following important environment variables:

- PATH - The PATH variable is extended by path to Tivoli Workload Scheduler binaries. The following directories are referenced:
  - *<TWS_HOME>*
  - *<TWS_HOME>*/bin
  - *<TWS_HOME>*/wastools
- TWS_TISDIR - This variable points to the Tivoli Workload Scheduler installation directory. It is used by the command-line interface utilities. It points to the directory containing *translation tables* that are used for displaying the messages in the correct language and codeset.

  If you have multiple Tivoli Workload Scheduler versions installed in the same system, make sure that this variable points to the newest version.
- UNISONHOME - This variable points to the Tivoli Workload Scheduler installation directory.

Setting up additional environment variables

This section describes how to set environment variables for obtaining optimal output from **`composer`** and **`conman`** commands.

You must set two important environment variables:

- MAESTROLINES=0 - Disables page breaks for **`composer`** or **`conman`** output. Output is written at once, without splitting output to pages and requiring an operator's confirmation.
- MAESTRO_OUTPTUT_STYLE=LONG - Prevents **`composer`** and **`conman`** output truncating. The output value is written at full length and is not truncated.

6.2 Switching the domain manager

In this section we describe the Tivoli Workload Scheduler failover. We focus on the following topics:

- "Terminology" on page 155
- "Domain manager crash impact" on page 156
- "Recovering from Tivoli Workload Scheduler failover" on page 156
- "Long-term switch and short-term switch" on page 157
- "Backup domain manager considerations" on page 158
- "Backup master domain manager considerations" on page 159
- "Determining suitable candidates" on page 159
- "Switching the domain manager" on page 164
- "Determining the current master domain manager" on page 167

6.2.1 Terminology

As discussed in Chapter 2, "Planning" on page 11, the Tivoli Workload Scheduler network is structured into *domains*. Each domain consists of agents (mostly fault-tolerant agents) and one domain manager. The top level Tivoli Workload Scheduler domain is managed by the master domain manager.

This section describes the impact of a domain manager crash. Further we discuss the Tivoli Workload Scheduler failover process - switching from the domain manager to the backup domain manager

Most of the following sections describe the domains in the Tivoli Workload Scheduler network.

In special cases, when referring to the top-level Tivoli Workload Scheduler domain, we emphasize that we are referring to the master domain manager and the backup master domain manager.

The process that manages failover from the domain manager to the backup domain manager is called *switch manager* (or *switchmgr*).

6.2.2 Domain manager crash impact

This section describes the impact of domain manager failure. A crash of a domain manager results in the following:

- Subordinate agents and domain managers are not able to resolve inter-workstation dependencies because they do not receive any information from the crashed domain manager.
- The information from the affected domains is not sent upwards. The “upper” agents and domains are not able to resolve dependencies from domains managed by the crashed domain manager.
- Standard agents hosted by the crashed domain manager stop processing because they are fully dependent on their hosting domain manager.

6.2.3 Recovering from Tivoli Workload Scheduler failover

This section describes how to recover from the domain manager crash.

If a domain manager crashes, it is possible to pass control to the backup domain manager. By default, this process is not automatic; you must invoke it. Certain prerequisites must be met to switch to the backup domain manager.

A different set of prerequisites is required for a long-term switch of the master domain manager, and a different set again is required in other cases.

We describe the prerequisites in the following sections.

6.2.4 Long-term switch and short-term switch

This section discusses the long term switch and the short term switch of the master domain manager. By *long-term* and *short-term* switches, we mean the following:

- Short term - In the event of master domain manager downtime (crash or maintenance), you can expect that the master domain manager is backed up and fully operational again by the start time of the FINAL job stream.

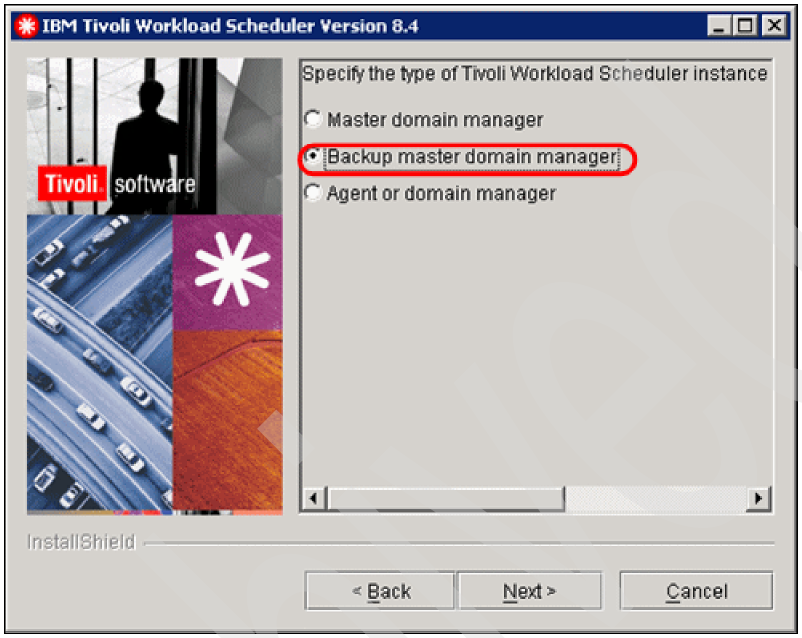
  In this case, you can switch to any suitable candidate (any fault-tolerant agent with full status that is a member of the top-level Tivoli Workload Scheduler domain). This short-term backup master domain manager assumes the plan-related duties of the master domain manager. However, it is unable to generate the new production plan because it does not have access to the database.

- Long term - In a long-term situation, you can expect downtime that is not resolved by the start time of the FINAL job stream.

  In this case, you must switch to the workstation that is installed as the backup master domain manager. This workstation has a full installation of the Tivoli Workload Scheduler engine including the WebSphere Application Server.

  The workstation installed as the backup master domain manager has the same capabilities as the master domain manager. Once the backup master domain manager becomes a manager of the top-level Tivoli Workload Scheduler domain, it can fully replace the original master domain manager.

Figure 6-1 shows a window from the installation process of a Tivoli Workload Scheduler workstation. Here you can determine which type of workstation you are installing.


IBM Tivoli Workload Scheduler Version 8.4

Specify the type of Tivoli Workload Scheduler instance

Master domain manager
[✓ Backup master domain manager]
Agent or domain manager

Figure 6-1 Installing the backup master domain manager

Note: A long-term switch to a fully capable backup master domain manager is necessary when you are sure that the FINAL job stream is not run on the master domain manager.

In such cases, it is necessary to switch to the backup master domain manager and not just to any fault-tolerant agent with full status.

6.2.5 Backup domain manager considerations

This section describes the necessary prerequisites for the following cases:

- Short-term switch of the master domain manager to the backup master domain manager
- Switch of any other domain manager to the backup domain manager

A suitable candidate for the backup domain manager must meet the following prerequisites:

- Must be a fault-tolerant agent with full status
- Must be member of the same domain as the original domain manager
- Must be able to establish two-way IP links (connections) to the subordinate workstations in the network

Note: If the node is not a fault-tolerant agent in a full status or is not in the same domain, it cannot act as a backup domain manager in that domain. In such cases, the **`switchmgr`** command fails.

6.2.6 Backup master domain manager considerations

This section describes the necessary prerequisites for a long-term switch to the backup master domain manager.

As stated previously, *long-term* in this context means that we do not expect the master domain manager to recover by the start time of the FINAL job stream. In this case, we perform steps that fully replace the master domain manager with the backup master domain manager. The backup master domain manager must be able to access the database, run the job stream FINAL, and build the production plan.

For these reasons, the requirements for the backup master domain manager are more strict:

- Must be a fault-tolerant agent in full status
- Must be member of the same domain as the master domain manager
- Must have access to the Tivoli Workload Scheduler database

Note: If the node is not a fault-tolerant agent with full status or is not in the top-level domain, it cannot act as a backup master domain manager. In this case the **`switchmgr`** command fails.

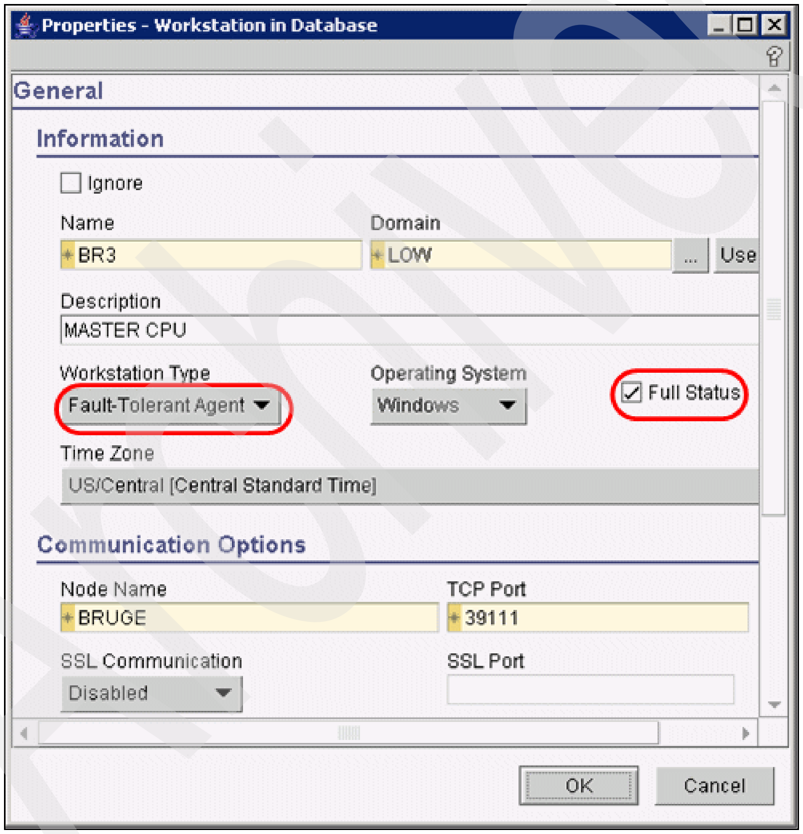
6.2.7 Determining suitable candidates

This section describes how to identify workstations that can act either as backup domain managers or as backup master domain managers. For example, you can check whether or not the workstation is a full status fault-tolerant agent. We describe the process of determining suitable candidates in the following sections.

Determining candidates using the Job Scheduling Console

Perform the following steps to check whether the workstation is a full status fault-tolerant agent:

1. Expand the **Default Database List** branch.
2. Select **All Workstations.**
3. From the list, double-click the desired workstation
4. Check the following settings (see Figure 6-2):
   – Workstation Type - Fault-Tolerant Agent
   – Full Status check box is selected.


Properties - Workstation in Database

General

Information

☐ Ignore

Name: BR3

Domain: LOW

Description: MASTER CPU

Workstation Type: Fault-Tolerant Agent

Operating System: Windows

☑ Full Status

Time Zone: US/Central [Central Standard Time]

Communication Options

Node Name: BRUGE

TCP Port: 39111

SSL Communication: Disabled

SSL Port:

Figure 6-2 Checking the Full Status option using Job Scheduling Console

Determining candidates using composer

Perform the following steps to check whether the workstation is a full status fault-tolerant agent:

1. Open a command-line window.
2. Make sure you have sourced the necessary Tivoli Workload Scheduler command-line environment.
3. Issue the following command:

   `composer display cpu=@`

4. The workstation definition must include the following entries:
   - TYPE FTA
   - FULLSTATUS ON

Example 6-1 shows the output of the **`composer`** command. The first workstation displayed in the list (BR1) is a suitable candidate for the domain manager while the second workstation (BR3) is not.

Example 6-1 Checking the Full Status option using composer

```
C:\>composer display cpu=@

CPUNAME BR1
  DESCRIPTION "MASTER CPU"
  OS WNT
  NODE BR1 TCPADDR 38111
  TIMEZONE US/Central
  DOMAIN LOW
  FOR MAESTRO
    TYPE FTA
    AUTOLINK ON
    BEHINDFIREWALL OFF
    FULLSTATUS ON
END

CPUNAME BR3
  DESCRIPTION "MASTER CPU"
  OS WNT
  NODE BR3 TCPADDR 39111
  TIMEZONE US/Central
  DOMAIN LOW
  FOR MAESTRO
    TYPE FTA
    AUTOLINK ON
    BEHINDFIREWALL OFF
    FULLSTATUS OFF
END
```

Determining full status using conman

Perform the following steps to check whether the workstation is a full status fault-tolerant agent:

1. Open a command-line window.
2. Make sure that you have sourced the necessary Tivoli Workload Scheduler command-line environment.
3. Issue the following command:

   ```
   conman showcpus;link
   ```

4. The workstation entry must include the `F` flag.

Example 6-2 shows the output of the **`conman`** command. The first workstation displayed in the list (BR1) is a suitable candidate for the domain manager while the second workstation (BR3) is not.

Example 6-2 Checking the Full Status option using conman

```
C:\>conman "showcpus;link"
%showcpus;link
CPUID     HOST    FLAGS  ADDR  NODE
BR1       BR1       AF T  37111 BR1
BR3       BR3       A  T  39111 BR3
```

Note: If you want to determine a suitable candidate using the **`conman`** command, you *must* supply the `link` argument. Running **`conman showcpus`** without arguments does not display whether the workstation has full status.

Getting more information about a workstation from plan

More information about the workstation is offered by the cpuinfo utility. This utility gives information similar to the **`composer`** command but with the following differences:

- **`composer`** displays information from the database.
- **`cpuinfo`** displays information about a workstation defined in the plan.

The disadvantage of using the cpuinfo utility is that you must explicitly supply the exact name of the workstation. This utility does not accept any wildcards.

Once you have determined the workstation name, perform the following steps:

1. Open a command-line window.
2. Make sure that you have sourced the necessary Tivoli Workload Scheduler command-line environment.
3. Issue the following command:

   `cpuinfo <workstation_name>`

Example 6-1 on page 161 shows the output of the **`cpuinfo`** command. This information is extracted from the current production plan (from the Symphony file).

Example 6-3 Getting workstation details from plan using the cpuinfo utility

```
C:\>cpuinfo BRUGE
OS_TYPE: WNT
NODE: BRUGE
PORT: 31111
SSLPORT: 0
SEC_LEVEL: NONE
AUTOLINK: ON
FULLSTATUS: ON
RESOLVEDEP: ON
BEHINDFIREWALL: OFF
HOST: BRUGE
DOMAIN: MASTERDM
METHOD:
SERVER:
TYPE: MASTER
TIME_ZONE: US/Central
VERSION: 8.4
INFO: Windows NT 3790 5.2 INTEL/Single Pro
```

Note: Even if the workstation information is similar to the composer output, the cpuinfo utility extracts the information from the plan, not from the database.

6.2.8 Switching the domain manager

This section demonstrates three possible ways of switching the domain manager to the backup domain manager.

Necessary privileges for switching the domain manager

The machine that becomes the new domain manager can be restarted during the switching process. Therefore you need following privileges for the backup domain manager:

- Start
- Stop

These privileges are set in the security file and are related to the workstation (CPU) that becomes the new domain manager.

Note: You need not be root or administrator to switch the domain manager. You need the correct definition in the security file enabling you to start and stop the backup domain manager.

Switching the manager from conman

This section describes how to switch the domain manager from the conman interface.

Perform the following steps:

1. Open a command-line window.
2. Make sure that you have sourced the necessary Tivoli Workload Scheduler command-line environment.
3. Issue the following command:

   ```
   conman switchmgr <domain>;<backup domain manager>
   ```

   The following two arguments must always be passed to the **conman switchmgr** command:

   – domain - Name of the domain whose domain manager you want to switch

   – backup domain manager - Name of the backup domain manager that acts as the new domain manager

   These two arguments must be separated by the semicolon character.

You can issue the **conman switchmgr** command on the following nodes in the Tivoli Workload Scheduler network:

- Any fault-tolerant agent in any domain

- Any domain manager of any domain
- Any backup domain manager in any domain

Note: When running the **`conman switchmgr`** command, you must supply the target domain as an argument. This means that you can run switchmgr in any domain, regardless of whether it affects the current domain or another one.

Switching the manager from the command line

This section describes how to switch the domain manager directly from the command line (Microsoft Windows or UNIX terminal).

Perform the following steps:

1. Log on to the backup domain manager (the workstation where you want to switch the domain manager).
2. Open a command-line window.
3. Make sure that you have sourced the necessary Tivoli Workload Scheduler command-line environment.
4. Issue the following command:

   ```
   switchmgr
   ```

When running the **`switchmgr`** command *outside* the conman interface, you are actually launching a script that performs the switch of the domain manager to the current workstation. Even if you specify the target domain and name of the backup domain manager, these parameters are ignored, and the domain manager is switched to the current workstation.

Note: When running the **`switchmgr`** script from the command line (not from the conman interface), you need not supply any arguments because you are switching the domain manager to the workstation where you are logged on.

Switching the manager from the Job Scheduling Console

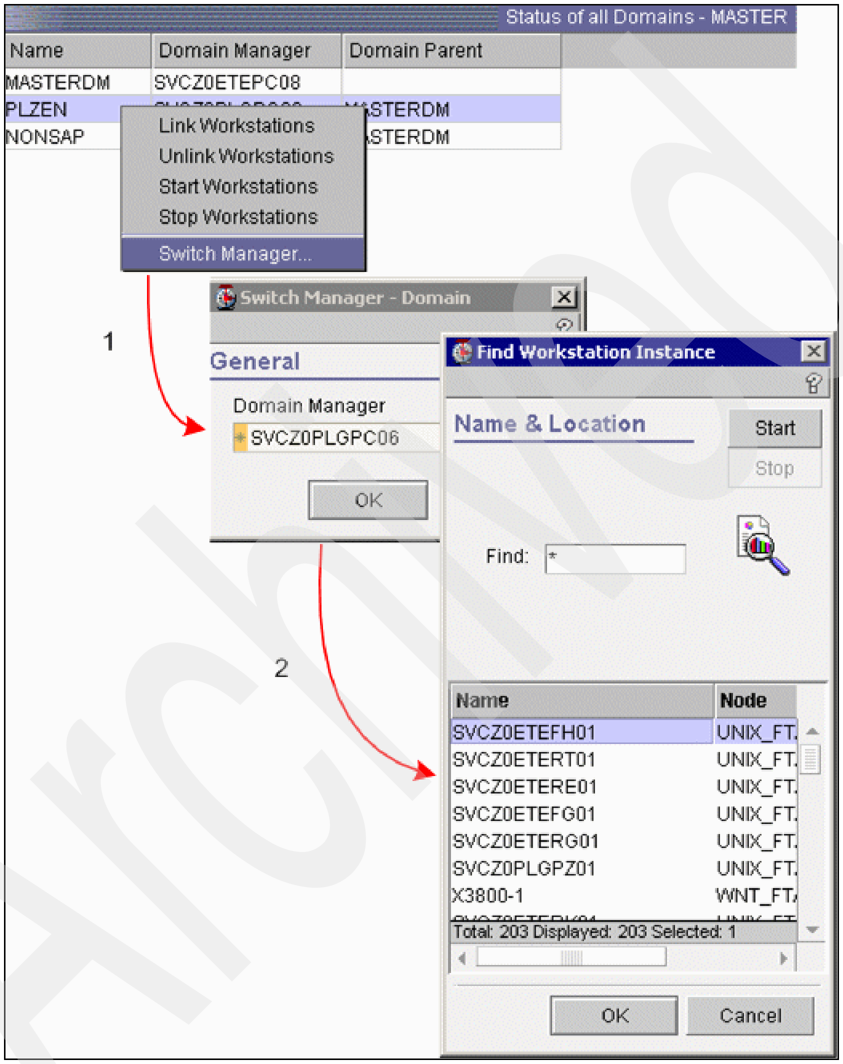
This section describes how to switch the domain manager from the Job Scheduling Console.

Perform the following steps:

1. Log on to the Job Scheduling Console.
2. In the Default Plan List, select **Status of all Domains**.
3. Right-click the desired domain and select **Switch Manager**.

4. Fill-in the name of the new domain manager directly or invoke a search window and select the new domain manager from the list.

See Figure 6-3 for more details.



Name	Domain Manager	Domain Parent
MASTERDM	SVCZ0ETEPC08	
PLZEN		MASTERDM
NONSAP		STERDM

Domain Manager: SVCZ0PLGPC06

Find: *

Name	Node
SVCZ0ETEFH01	UNIX_FT
SVCZ0ETERT01	UNIX_FT
SVCZ0ETERE01	UNIX_FT
SVCZ0ETEFG01	UNIX_FT
SVCZ0ETERG01	UNIX_FT
SVCZ0PLGPZ01	UNIX_FT
X3800-1	WNT_FT

Total: 203 Displayed: 203 Selected: 1

Figure 6-3 Switching the domain manager using Job Scheduling Console

6.2.9 Determining the current master domain manager

This section describes how to determine the currently acting master domain manager. If you want to determine the currently acting master domain manager, or if you want to check whether the domain manager has been switched successfully, perform the following:

1. Log on to the machine which is member of the top-level Tivoli Workload Scheduler domain (usually MASTERDM).
2. Open a command-line window.
3. Make sure that you have sourced the necessary Tivoli Workload Scheduler command-line environment.
4. Issue the following command:

   `conman showcpus`
5. The currently acting master domain manager is identified by the term MASTER in the node field.

Example 6-4 shows the output of the **`conman showcpus`** command.

Example 6-4 Determining the MASTER using conman

```
%showcpus
CPUID      RUN NODE        LIMIT FENCE DATE     TIME  STATE
BRUGE       48 *WNT  MASTER   60     0 05/07/08 12:12    I J
ATHENS      48  WNT  FTA       0     0 05/07/08 12:11  LTI JW
BR1         48  WNT  FTA       0     0 05/07/08 12:12  LTI JW
BR2         48  WNT  MANAGER   0     0 05/07/08 12:12  LTI JW
```

Note: The asterisk sign (*) does not identify the current domain manager. It simply indicates that you are running the **`conman`** command from that machine. You can identify the current master domain manager only by the flag MASTER in the node field.

6.3 Backup and restore tasks

This section describes how to perform regular backups of the Tivoli Workload Scheduler environment. We focus on the following topics:

- "Scheduling objects in the Tivoli Workload Scheduler database" on page 168
- "Back up scheduling objects to text files" on page 168

- "Restoring the scheduling objects from text files" on page 169
- "Backing up using DB2 tools" on page 169
- "Backing up and restoring local parameters" on page 170
- "Backing up and restoring WebSphere Application Server settings" on page 171

6.3.1 Scheduling objects in the Tivoli Workload Scheduler database

The Tivoli Workload Scheduler database contains definitions of all scheduling objects:

- Jobs
- Job streams
- Microsoft Windows users
- Prompts
- Resources
- Global parameters
- Calendars
- Workstations, workstation classes, and domains

Note: Local parameters are defined extra for each workstation. Local parameters are not stored in the Tivoli Workload Scheduler database.

6.3.2 Back up scheduling objects to text files

This section describes how to extract scheduling objects from the Tivoli Workload Scheduler database so that they can be backed up as files.

The following commands are used for extracting the Tivoli Workload Scheduler scheduling objects from the database to text files:

- **`composer create`** `<`*`file`*`> from <`*`scheduling_object_selector`*`>`
- **`composer extract`** `<`*`file`*`> from <`*`scheduling_object_selector`*`>`

  The commands **`composer create`** and **`composer extract`** are equivalent.

Important: The preceding commands work with the Tivoli Workload Scheduler database only. You cannot backup local parameters by using these commands.

6.3.3 Restoring the scheduling objects from text files

This section describes how to import scheduling objects from text files to the Tivoli Workload Scheduler database. The following commands are used to do this:

- **`composer add`** *<file>*
- **`composer replace`** *<file>*

> **Note:** Commands **`composer add`** and **`composer replace`** are similar but not equivalent. See 5.1.6, “Working with composer” on page 111 for details.

6.3.4 Backing up using DB2 tools

This section describes how to back up the Tivoli Workload Scheduler database using the DB2 command line.

Tivoli Workload Scheduler V8.4 uses a relational database for persistent storage. When referring to the *Tivoli Workload Scheduler database*, we mean a complete set of the scheduling objects. This database is physically represented by a set of tables in the relational database.

The following databases are supported in Tivoli Workload Scheduler V8.4:

- DB2
- Oracle

This section describes how to back up the Tivoli Workload Scheduler database when using DB2 and the default name for the Tivoli Workload Scheduler database.

Perform the following steps:

1. Log on to the master domain manager.
2. Open a command-line window.
3. Make sure that you have sourced the necessary Tivoli Workload Scheduler command-line environment.
4. Before running the backup you must stop any application connected to the database. By default only the WebSphere Application Server on the master domain manager should be connected to the database. Stop the WebSphere Application Server instance by issuing the following command:

   ```
   conman "stopappserver;wait"
   ```

5. Make sure that you have sourced the necessary DB2 command-line environment.
6. Back up the database defined for Tivoli Workload Scheduler. The default name of this database is TWS. Issue the following command:

   ```
   db2 backup database TWS
   ```

Important: Database backup does not contain the local parameters definition.

6.3.5 Backing up and restoring the security file

This section describes how to back up and restore the Tivoli Workload Scheduler security file.

The following command is used for dumping the security file to the text file:

```
dumpsec > <file>
```

Note: You must specify the redirect (>) character. The `dumpsec` command by default writes to the standard output. You must redirect the output to a file to perform backups or modifications.

The following command is used for importing the security entries from the text file:

```
makesec <file>
```

Note: The `makesec` command must not specify the redirect ("<") character. Supply the file name as the only argument.

6.3.6 Backing up and restoring local parameters

This section describes how to perform backup and restore of local parameters. Local parameters are not part of the Tivoli Workload Scheduler database, and therefore, they cannot be backed up together with scheduling objects.

Local parameters can be managed using the command-line interface utility parms. Since the release of Tivoli Workload Scheduler V8.4, this utility has been extended with backup and restore features.

The following command is used for extracting of the local parameters to a text file:

```
parms -e <file>
```

The following command is used for importing the local parameters from a text file:

```
parms -r <file>
```

6.3.7 Backing up and restoring WebSphere Application Server settings

This section describes how to perform backup and restore of the WebSphere Application Server configuration.

Perform the following steps:

1. Log on to the master domain manager.
2. Open a command-line window.
3. Make sure that you have sourced the necessary Tivoli Workload Scheduler command-line environment.
4. To back up the WebSphere Application Server configuration, issue the following command:

   ```
   backupConfig <file>
   ```

> **Note:** `backupConfig` and `restoreConfig` scripts or commands are placed in:<*twshome*>/wastools.

5. To restore the WebSphere Application Server configuration, issue the following command:

   ```
   restoreConfig <file>
   ```

> **Note:** The WebSphere Application Server needs to be restarted after the `restoreConfig` command.

6.4 Administering the WebSphere Application Server

Chapter 4, “Configuration” on page 45 describes in detail how to perform administrative tasks necessary for changing the WebSphere Application Server configuration settings.

The WebSphere Application Server hosts important J2EE™ applications that run on the master domain manager. It also provides connectivity to the database and hosts the connector either from the command line or from the Job Scheduling Console.

The WebSphere Application Server instance is installed on the following nodes:

- Tivoli Workload Scheduler master domain manager
- Tivoli Workload Scheduler backup master domain manager

It is important to configure particular settings of the WebSphere Application Server. Refer to 4.6, “Configuring WebSphere Application Server” on page 83 for descriptions of how to view and modify the current WebSphere Application Server configuration.

6.5 Administering the DB2 database

Tivoli Workload Scheduler V8.4 provides a set of scripts enabling you to administer and maintain the database used for persistent data storage.

The database tools are located on the master domain manager in the following directories:

- *<TWS_HOME>*/dbtools - General directory containing the database scripts and SQL statements
- *<TWS_HOME>*/dbtools/db2/scripts - DB2 tools
- *<TWS_HOME>*/dbtools/oracle/scripts - Oracle tools

Important DB2 administrative tools

This section describes important DB2 administrative tools that help you maximize performance of the database. You must have administrative access to the database to use these tools. You must source the necessary database command-line environment as well.

You must run the following tools periodically to maximize database performance:

- dbreorg - This tool physically reorganizes the data tables and indexes in the database and thus optimizes disk space usage and ease of data access.

  We recommend you perform the following steps before running the dbreorg script:

  a. Back up the Tivoli Workload Scheduler database.

  b. Stop *all* Tivoli Workload Scheduler processes including the WebSphere Application Server.

Note: Dbreorg cannot run successfully when the database is being accessed. To avoid unpredictable results, stop all Tivoli Workload Scheduler processes including WebSphere Application Server before running dbreorg.

- dbrunstats - Calculates and sets the performance parameters of the database so that performance can be maximized.

Note: It is essential to understand that these tools must be issued after recovery from Symphony file corruption. If you had to scratch the current production plan (using the `ResetPlan -scratch` command), your next step should be to launch dbrunstats.

The current Tivoli Workload Scheduler documentation recommends running dbreorg after executing `ResetPlan -scratch`. This recommendation is not correct. Only dbrunstats should be run after `ResetPlan -scratch`.

6.6 Administering event-driven workload automation

Tivoli Workload Scheduler was originally designed to automate the execution of a batch workload according to predefined calendars. While calendar-based scheduling in Tivoli Workload Scheduler was a powerful feature, it was not flexible when reacting to asynchronous events occurring in the environment.

Tivoli Workload Scheduler V8.4 introduces the concept of *event-driven workload automation* (EDWA). This powerful and flexible feature allows Tivoli Workload Scheduler to react to asynchronous events, regardless of whether they occurred within or outside the Tivoli Workload Scheduler environment.

Tivoli Workload Scheduler can react to incoming events, then launch user-defined actions. The logic that evaluates incoming events and connects them with actions is computed by event rules.

Note: A detailed description of event-driven workload automation is provided in *Deployment Guide Series: IBM Tivoli Workload Scheduler V8.4 and IBM Tivoli Dynamic Workload Broker V1.2*, SG24-7528.

6.6.1 Event-driven workload automation concept

The concept of event-driven workload automation is based on events, event rules, and actions. Events represent the conditions that occur either inside or outside the Tivoli Workload Scheduler environment. You respond to an event with the appropriate action determined by the nature of the incoming event. You define event rules, which is the logic that links events to actions.

Figure 6-4 demonstrates the concept of event-driven workload automation.

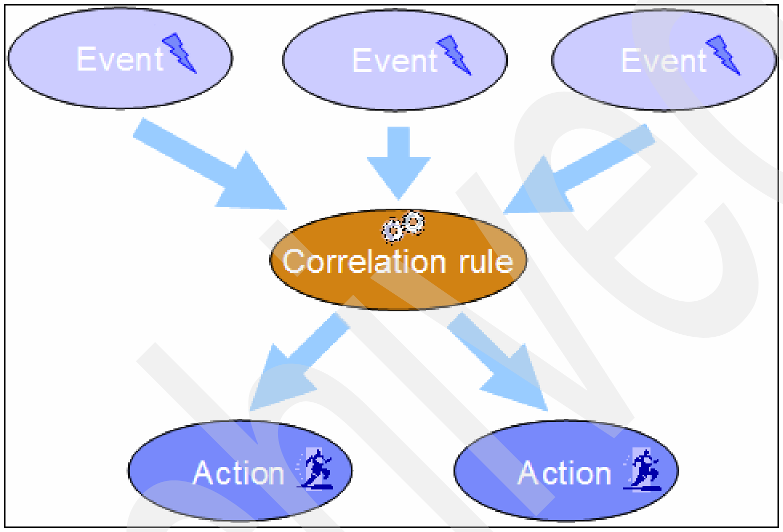


Figure 6-4 Event-driven workload automation

6.6.2 Event rules

An event rule defines which actions should be triggered based on incoming events. More specifically, the event rule acts as a logical connector between events (inputs) and actions (outputs).

An event rule definition consists of the following:

- Events
- Event correlation condition
- Actions

Note: An event rule specifies the correlation condition that evaluates the incoming events and launches consecutive actions.

6.6.3 Built-in events

This section describes built-in events recognized by the Tivoli Workload Scheduler V8.4. Built-in events determine which conditions can be monitored either inside or outside the Tivoli Workload Scheduler environment.

In general, we can classify built-in events as follows:

- External events related to file monitoring - Monitoring agents can monitor file system conditions from various perspectives. Typical examples of generated events are the following:
  - Log message written
  - File created
  - File deleted
  - Modification completed
- Tivoli Workload Scheduler plan events - Events are generated during production plan processing. You can define the scope of monitored scheduling objects either explicitly or by using wildcards. Typical examples of generated events are the following:
  - Job Status Changed
  - Job Stream Status Changed
  - Job Stream Completed
  - Job Stream Cancelled
  - Job Stream Late
  - Workstation Status Changed
  - Child Workstation Link Changed
  - Prompt Status Changed

Note: A complete list and description of built-in events is provided in *Deployment Guide Series: IBM Tivoli Workload Scheduler V8.4 and IBM Tivoli Dynamic Workload Broker V1.2*, SG24-7528.

6.6.4 Built-in actions

This section describes the built-in actions provided with the Tivoli Workload Scheduler V8.4. Actions specify the tasks that should be performed when the incoming event satisfies the correlation condition defined within the event rule.

In general, we can classify built-in actions as follows:

- Operational actions - These actions perform operative tasks within the Tivoli Workload Scheduler environment. Typical examples of operational actions are the following:
  - Submit a job with the name BACKUP_TSM_DB
  - Submit a job stream with the name PERFORM_CLEANUP
  - Reply to Prompt1 prompt

  These actions are related to the Tivoli Workload Scheduler environment only.
- Notification actions - These actions notify either you or applications about particular conditions that have occurred in your environment. Typical examples of notification actions are the following:
  - Send an e-mail with defined body and subject to the recipients
  - Log a message in an internal audit database
  - Forward an event to Tivoli Enterprise Console
- Generic actions - These types of actions are used for performing generic tasks. Typical examples of notification actions are the following:
  - Run an operating system command on the event processor. (See “Event-driven workload automation topology” on page 176 for a description of this component.)
  - Run a script on the event processor.

Note: Notification actions send information about events in the Tivoli Workload Scheduler environment. Operational actions can submit new workloads (jobs, job streams, and ad hoc jobs) and reply to prompts.

Tivoli Workload Scheduler V8.4 does not provide built-in actions for updating an already submitted workload.

6.6.5 Event-driven workload automation topology

This section describes the topology of event-driven workload automation. We highlight the most important components and their purposes.

From the topological perspective, the event-driven workload automation can be divided into the following components:

- Event processor - Central point of the event-driven workload automation topology. Event rules running within the event processor are deployed from here to monitoring agents.

- Monitoring agents - Locally monitor events based on the rules that have been deployed to them from the event processor.
- Database - Event-driven workload automation uses part of the Tivoli Workload Scheduler database.
- User interfaces - Event-driven workload automation can be managed from the command-line interface or from the Tivoli Dynamic Workload Console.

Figure 6-5 shows the event-driven workload automation topology.

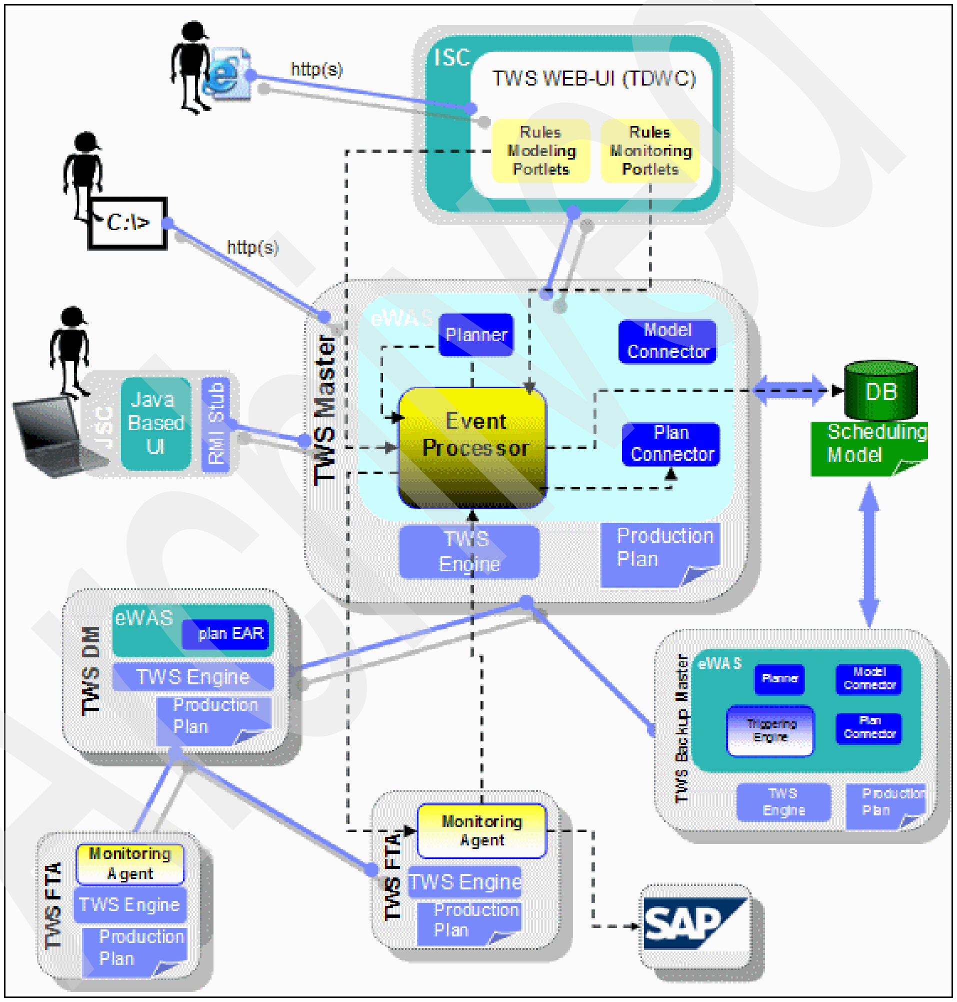


Figure 6-5 Event-driven workload automation - topological view

We have used the following abbreviations in Figure 6-5 on page 177:

- TWS - Tivoli Workload Scheduler
- TWS DM - Tivoli Workload Scheduler domain manager
- TWS FTA - Tivoli Workload Scheduler fault-tolerant agent
- JSC - Job Scheduling Console
- eWAS - Embedded version of WebSphere Application Server
- ISC - Integration Solutions Console
- TWS WEB-UI - Tivoli Workload Scheduler Web user interface, which is another name for the Tivoli Dynamic Workload Console

  The Job Scheduling Console by itself does not provide an interface for interaction with event-driven workload automation. It launches a Web browser pointing to the Tivoli Dynamic Workload Console.

6.6.6 Administering event-driven workload automation

This section describes basic administrative tasks related to event-driven workload automation. We focus on the following topics:

- "Starting and stopping the event processor"
- "Starting and stopping monitoring agents"
- "Deploying event rules"

Starting and stopping the event processor

The event processor is a J2EE application running in WebSphere Application Server. It can run either on the master domain manager or on the backup master domain manager.

The following commands are used for starting and stopping the event processor:

- **`conman starteventprocessor`** - Starts the event processor.
- **`conman stopteventprocessor`** - Stops the event processor.

These commands are issued from any workstation in the Tivoli Workload Scheduler network.

Note: No space character is included in the `starteventprocessor` or `stopteventprocessor` arguments.

Starting and stopping monitoring agents

By default, monitoring agents are deployed to all fault-tolerant agents. They are started automatically by default. If you do not want to start monitoring agents automatically, you must set the following value in the localopts file:

```
autostart monman=no
```

The following commands can be used for starting and stopping the monitoring agent:

- **`conman startmon`** - Starts the monitoring agent
- **`conman stopmon`** - Stops the monitoring agent

If you do not specify any additional arguments, the monitoring processes starts or stops on the local workstation.

Deploying event rules

This section describes the deployment part of the event rule life cycle. We provide an explanation of how the event rule gets activated so that the events can be generated.

Deploying event rules from the event processor

This section describes how the event processor invokes rule deployment to workstations. Follow these steps to start monitoring a particular condition:

1. Create the event rule in the rule editor and save the rule definition.
2. Set the rule as complete. This state determines that the rule development is finished, and the rule can be deployed to monitoring agents.
3. Distribute the rule to monitoring agents. This can be done either automatically or manually:
   - Automatic deployment is performed periodically. It is dependent on the value of the global option `deploymentFrequency` (`df`). By default the period is set to five minutes. If you set this value to zero, the automatic deployment is switched off.
   - Manual deployment can be accomplished by issuing the **`planman deploy`** command.

Requesting the current rules definition on a workstation

This section describes how a particular workstation requests receipt of the current rules from the event processor.

If you are logged on to a particular workstation and want to receive the most current version of the event rules, issue the following command:

```
conman deployconf
```

If you do not specify additional arguments, the event rules are deployed to the local workstation.

Starting all components from a workstation

This section describes how to start all important event-driven workload automation components on the workstation, so that the workstation has the most current event rules definition.

In this section, we assume that all components of event-driven workload automation are stopped.

Perform the following steps:

1. Log on to the workstation.
2. Source the Tivoli Workload Scheduler command-line interface environment.
3. Issue the following commands:
   a. **`conman deployconf`** - To receive the most current event rule definitions
   b. **`conman starteventprocessor`** - To remotely start the event processor running on the master domain manager (or backup master domain manager)
   c. **`conman startmon`** - To start the local monitoring agent

6.7 Administering workstations

In this section we describe the common tasks related to workstation administration and maintenance. We focus on the following topics:

- "Starting and stopping workstations" on page 181
- "Maintaining a workstation's file systems" on page 198

6.7.1 Starting and stopping workstations

This section describes the steps we recommend you perform when starting and stopping Tivoli Workload Scheduler workstations. We describe the necessary steps for the following platforms:

- UNIX and Linux platforms
- Microsoft Windows platform

We cover fault-tolerant agents and the master domain manager in separate sections.

UNIX and Linux platforms

This section describes the startup and shutdown procedures on UNIX and Linux workstations.

Tivoli Workload Scheduler components on UNIX and Linux platforms are represented as *processes*. The following processes are present on UNIX and Linux platforms:

- netman

Note: The netman configuration file exists on all Tivoli Workload Scheduler workstations to define the services provided by netman. It is called <*TWS_home*>/network/Netconf.

- mailman
- batchman
- jobman
- writer
- monman

We cover starting up and shutting down workstations (fault-tolerant agents and master domain manager) in separate sections.

Startup procedure on UNIX fault-tolerant agent

This section describes the startup procedure on UNIX and Linux fault-tolerant agents.

The first process that you must start is netman. This process can be started by the following script:

```
<TWS_HOME>/StartUp
```

Note: This program should be run under the <*TWSuser*> - owner of the Tivoli Workload Scheduler instance. You should not run the StartUp script under the root user.

The StartUp script launches these commands:

- **`<TWS_HOME>/bin/netman`** - Starts the netman process
- **`<TWS_HOME>/trace/init_trace`** - Starts the autotrace utility

By running the StartUp script, you start the netman process. To start the other Tivoli Workload Scheduler processes, you must perform one of the following tasks:

- Link and start the workstation from another workstation in the Tivoli Workload Scheduler network (mostly the workstation's domain manager).

  Issue the following commands:

  - **`conman`** `"link cpu=`*`<domain>`*`!`*`<workstation>`*`"`
  - **`conman`** `"start cpu=`*`<domain>`*`!`*`<workstation>`*`"`

- Start the workstation locally. Issue the following command:

  `conman start`

  The locally executed **`conman start`** command causes spawning of the following Tivoli Workload Scheduler processes on the affected fault-tolerant agent:

  - mailman
  - batchman
  - jobman

  The following processes are started separately:

  - monman - This process starts in the following cases:
    - Automatically when the machine receives a new copy of the Symphony file if the following local option is set:

      `autostart monman=yes`

    - Manually by issuing the following command:

      `conman startmon`

  - writer - This process is spawned by netman, when a link with a remote workstation is established.

Note: Only the netman process is started by the `StartUp script`. The other processes are spawned by `conman start -` either locally or remotely. When the workstation is linked, the writer process is also spawned.

Startup procedure of UNIX master domain manager

This section describes the startup procedure of the UNIX and Linux master domain manager or backup master domain manager.

The startup procedure of the master domain manager is exactly the same as on any other fault-tolerant agent. The only difference is in the content of the StartUp script. An additional command is issued in the StartUp script:

`<TWS_HOME>/wastools/startWas.sh` - Starts WebSphere Application Server

Note: The only difference in the startup procedure on the master domain manager is starting the WebSphere Application Server. The WebSphere Application Server is started from the StartUp script.

Shutdown of UNIX fault-tolerant agent

This section describes the shutdown procedure on the UNIX and Linux fault-tolerant agent.

Shutting down the UNIX fault-tolerant agent consists of the following steps:

1. Unlinking the workstation.
2. Stopping all Tivoli Workload Scheduler processes *except* netman and monman.
3. Stopping netman and monman.

To initiate these steps, issue the following commands on the workstation:

- **`conman unlink cpu=<workstations_domain manager>`** - To unlink from Tivoli Workload Scheduler network and stop the writer process
- **`conman stop;wait`** - To stop all processes *except* netman
- **`conman shut;wait`** - To stop netman and monman

Note: The **`conman stop`** command does not stop the netman and monman processes. By shutting down netman, you also stop monman.

Shutdown of UNIX master domain manager

This section describes the shutdown procedure of the UNIX and Linux master domain manager or backup master domain manager.

The shutdown procedure of the master domain manager is exactly the same as on any fault-tolerant agent. The only difference is that you must also shut down the WebSphere Application Server. An additional command must be issued to do so:

- **`conman "stopappserver;wait"`**
- **`conman "stop;wait"`**
- **`conman "shut;wait"`**

Remember to include the quotes character (") because on UNIX, the semicolon (;) terminates the command.

Note: The only difference in the shutdown procedure on the master domain manager is stopping the WebSphere Application Server. The WebSphere Application Server is stopped by the **`conman stopappserver`** command.

You can also concatenate multiple **`conman`** commands in one line:

- **`conman "stopappserver;wait & stop;wait"`** - Stops all Tivoli Workload Scheduler processes *except* netman and monman. WebSphere Application Server is also stopped.
- **`conman "stopappserver;wait & stop;wait & shut;wait"`**- Stops *all* Tivoli Workload Scheduler processes. WebSphere Application Server is also stopped.

Important ownership notices on UNIX platforms

This section contains important notices regarding the ownership of files and processes. The following list describes the correct ownership of Tivoli Workload Scheduler processes and binaries:

- All Tivoli Workload Scheduler processes on the workstation must run under the account of *<TWSuser>* except for the following:
  – jobman - Runs under the root account
  – WebSphere Application Server - Runs under the root account

- Most of the files in the <*TWS_HOME*>/bin directory must be owned by the <*TWSuser*> account. However, the following files must be owned by the root account:
  - jobman
  - starter
  - init.ssmagent
  - ssmagentstart
  - wasstart

Microsoft Windows platform

This section describes the startup and shutdown procedures on Microsoft Windows workstations. Tivoli Workload Scheduler components on Windows platforms are represented as processes and Windows services. The following processes are present on Windows platforms:

- netman.exe
- mailman.exe
- batchman.exe
- jobman.exe
- jobmon.exe
- writer.exe
- monman.exe
- tokensrv.exe

Tivoli Workload Scheduler components can be started either using the command-line interface or by starting the defined Windows services.

We cover starting up and shutting down fault-tolerant agents and the master domain manager in separate sections.

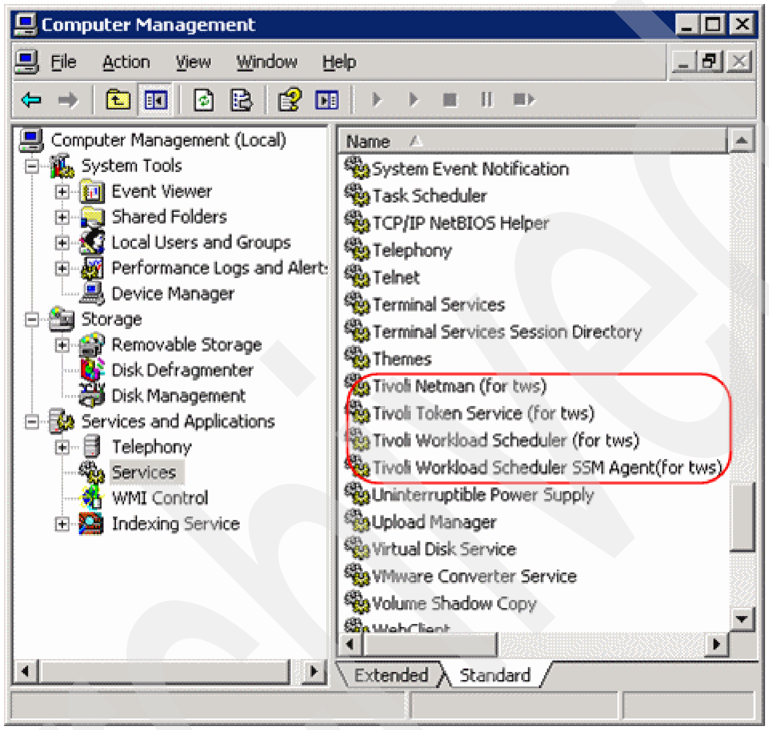
Startup procedure of Microsoft Windows fault-tolerant agent

By default the startup procedure on Microsoft Windows platforms is based on automatic starting of Windows services. During installation, the Tivoli Workload Scheduler creates the following set of services for each workstation instance:

- Tivoli Token service
- Tivoli Workload Scheduler service

- Tivoli Netman service
- Tivoli Workload Scheduler for SSM Agent (this is the service for the monitoring agent)

Figure 6-6 shows the services on the Microsoft Windows fault-tolerant agent.



Computer Management (Local)
- System Tools
  - Event Viewer
  - Shared Folders
  - Local Users and Groups
  - Performance Logs and Alert:
  - Device Manager
- Storage
  - Removable Storage
  - Disk Defragmenter
  - Disk Management
- Services and Applications
  - Telephony
  - Services
  - WMI Control
  - Indexing Service

System Event Notification
Task Scheduler
TCP/IP NetBIOS Helper
Telephony
Telnet
Terminal Services
Terminal Services Session Directory
Themes
Tivoli Netman (for tws)
Tivoli Token Service (for tws)
Tivoli Workload Scheduler (for tws)
Tivoli Workload Scheduler SSM Agent(for tws)
Uninterruptible Power Supply
Upload Manager
Virtual Disk Service
VMware Converter Service
Volume Shadow Copy
WebClient

Figure 6-6 Tivoli Workload Scheduler services

The following startup sequence is set up for the services:

1. By default all services start automatically (during boot up of the operating system).
2. Tivoli Workload Scheduler service is dependent on the start of the Tivoli Token service.
3. No other inter-service dependencies exist.

Figure 6-7 shows the corresponding Tivoli Workload Scheduler processes on a Microsoft Windows workstation.

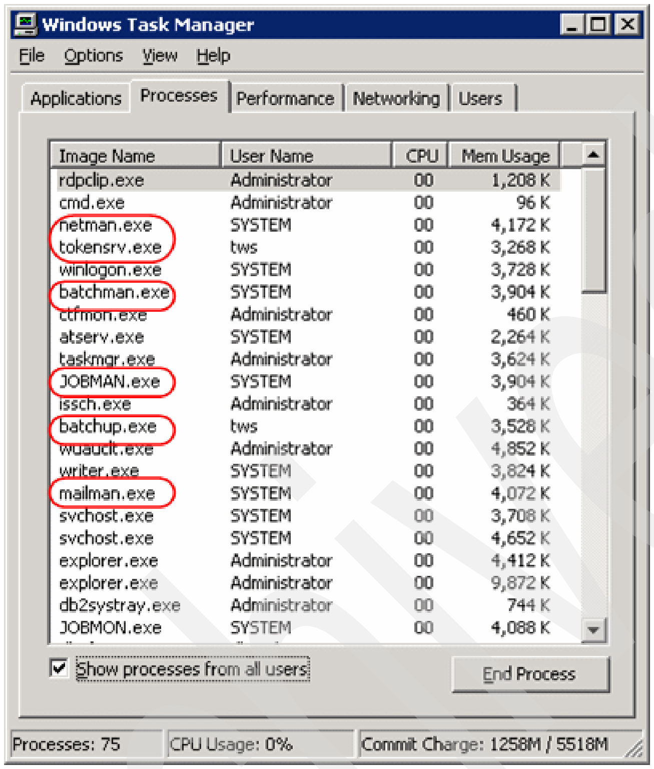

Windows Task Manager

Image Name	User Name	CPU	Mem Usage
rdpclip.exe	Administrator	00	1,208 K
cmd.exe	Administrator	00	96 K
netman.exe	SYSTEM	00	4,172 K
tokensrv.exe	tws	00	3,268 K
winlogon.exe	SYSTEM	00	3,728 K
batchman.exe	SYSTEM	00	3,904 K
ctfmon.exe	Administrator	00	460 K
atserv.exe	SYSTEM	00	2,264 K
taskmgr.exe	Administrator	00	3,624 K
JOBMAN.exe	SYSTEM	00	3,904 K
issch.exe	Administrator	00	364 K
batchup.exe	tws	00	3,528 K
wuauclt.exe	Administrator	00	4,852 K
writer.exe	SYSTEM	00	3,824 K
mailman.exe	SYSTEM	00	4,072 K
svchost.exe	SYSTEM	00	3,708 K
svchost.exe	SYSTEM	00	4,652 K
explorer.exe	Administrator	00	4,412 K
explorer.exe	Administrator	00	9,872 K
db2systray.exe	Administrator	00	744 K
JOBMON.exe	SYSTEM	00	4,088 K

☑ Show processes from all users

Processes: 75 CPU Usage: 0% Commit Charge: 1258M / 5518M

Figure 6-7 Tivoli Workload Scheduler processes on Windows workstation

Tivoli Workload Scheduler processes can be started either by Service Control Manager (which starts all services) or by using the StartUp.cmd script.

The StartUp.cmd script starts following services:

- autotrace
- Tivoli Netman Service
- Tivoli Token Service

By running the StartUp script, you start the Tivoli Token service, Tivoli Netman service, and autotrace only. To start the other Tivoli Workload Scheduler processes, you must do one of the following tasks:

- Link and start the workstation from another workstation in the Tivoli Workload Scheduler network (mostly the workstation's domain manager). Issue the following commands:
  - `conman "link cpu=<domain>!<workstation>"`
  - `conman "start cpu=<domain>!<workstation>"`
- Start the workstation locally. Issue the following command:

  ```
  conman start
  ```

The preceding commands cause spawning of the following Tivoli Workload Scheduler processes on the affected workstation:

- mailman
- batchman
- jobman
- JOBMON
- monman
- writer - This process is spawned by netman when a link with a remote workstation is established.

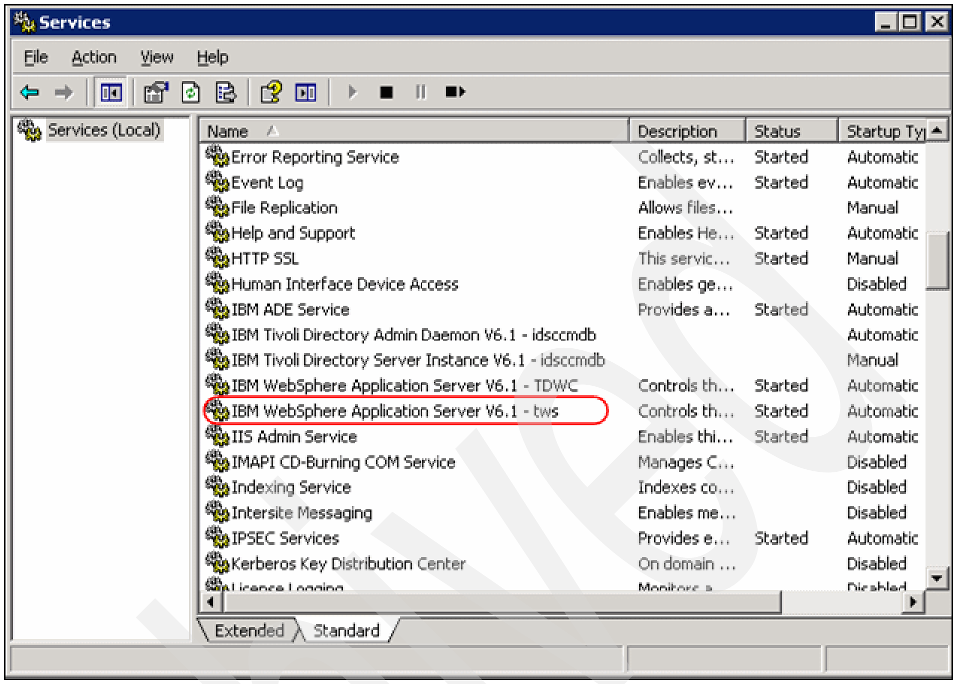
> **Note:** Only netman and tokensrv processes are started by the StartUp.cmd script. The other processes are spawned by **`conman start -`** either locally or remotely. When the workstation is linked, the writer process is also spawned.

Startup procedure of Microsoft Windows master domain manager

The startup procedure of the master domain manager is exactly the same as on any fault-tolerant agent. The only difference is that the startup procedure includes also starting the WebSphere Application Server. This is accomplished by the following service in Automatic startup status:

IBM WebSphere Application Server 6.1 - *<TWSuser>*

Figure 6-8 shows the service for the WebSphere Application Server.



Name	Description	Status	Startup Ty
Error Reporting Service	Collects, st...	Started	Automatic
Event Log	Enables ev...	Started	Automatic
File Replication	Allows files...		Manual
Help and Support	Enables He...	Started	Automatic
HTTP SSL	This servic...	Started	Manual
Human Interface Device Access	Enables ge...		Disabled
IBM ADE Service	Provides a...	Started	Automatic
IBM Tivoli Directory Admin Daemon V6.1 - idsccmdb			Automatic
IBM Tivoli Directory Server Instance V6.1 - idsccmdb			Manual
IBM WebSphere Application Server V6.1 - TDWC	Controls th...	Started	Automatic
IBM WebSphere Application Server V6.1 - tws	Controls th...	Started	Automatic
IIS Admin Service	Enables thi...	Started	Automatic
IMAPI CD-Burning COM Service	Manages C...		Disabled
Indexing Service	Indexes co...		Disabled
Intersite Messaging	Enables me...		Disabled
IPSEC Services	Provides e...	Started	Automatic
Kerberos Key Distribution Center	On domain ...		Disabled
License Logging	Monitors a...		Disabled

Figure 6-8 Service for the WebSphere Application Server on the master

The WebSphere Application Server also can be started manually. If you invoke the script StartUp.cmd, it includes also the following line:

```
conman "startappserver; wait"
```

Note: The only difference in the startup procedure on the master domain manager is starting the WebSphere Application Server. The WebSphere Application Server is started either automatically (as an automatic Windows service) or manually from the StartUp.cmd script.

Shutdown of the Microsoft Windows fault-tolerant agent

Shutting down a Microsoft Windows workstation consists of the following steps:

1. Unlinking the workstation.
2. Stopping the monitoring agent.
3. Stopping all Tivoli Workload Scheduler processes *except* netman and tokensrv.
4. Stopping netman and tokensrv.

To initiate these actions, issue the following commands on the workstation:

1. **`conman unlink cpu=<workstations_domain manager>`** - To unlink from the Tivoli Workload Scheduler network and stop the writer process
2. **`conman stopmon`** - To stop the monitoring agent
3. **`conman stop;wait`** - To stop all processes *except* netman and tokensrv
4. **`<TWS_HOME>/Shutdown.cmd`** - to stop Tivoli Netman service and Tivoli Token service

Note: The **`conman stop`** command does not stop the Tivoli Netman service. Furthermore, the monitoring agent must be stopped separately by issuing **`conman stopmon`**.

Shutdown of the Microsoft Windows master domain manager

The shutdown procedure of the master domain manager is exactly the same as on any fault-tolerant agent. The only difference is that you also must shut down the WebSphere Application Server. An additional command must be issued to do this:

- **`conman stopappserver;wait`**
- **`conman stop;wait`**
- **`conman shut;wait`**

Note: The only difference in the shutdown procedure on the master domain manager is stopping the WebSphere Application Server. The WebSphere Application Server is stopped by the **`conman stopappserver`** command.

The other way to shut down the WebSphere Application Server is to stop the corresponding Microsoft Windows service.

Important ownership notices on Microsoft Windows platform

This section contains important notices regarding ownership of Microsoft Windows services. The Windows services defined for Tivoli Workload Scheduler must have the following ownership:

- Tivoli Token service - Must be owned by *<TWSuser>*
- Tivoli Workload Scheduler service - Must be owned by *<TWSuser>*
- Netman service - Must be owned by LocalSystem
- Tivoli Workload Scheduler for SSM Agent - Must be owned by *<TWSuser>*

6.7.2 Checking workstation status

This section describes the steps necessary for checking whether a workstation is running properly and linked to its domain manager. We describe the steps common for UNIX, Linux, and Microsoft Windows platforms together. When different steps are required, we provide a separate section for each platform.

We describe following checks:

- “Checking local processes on UNIX and Linux workstations”
- “Checking status locally on Microsoft Windows workstations”
- “Checking Tivoli Workload Scheduler status”
- “Checking proper flags in showcpus”
- “Checking network communication ports”

Checking local processes on UNIX and Linux workstations

This section describes how to check whether the Tivoli Workload Scheduler processes on UNIX and Linux workstation are running. Issue the following command locally on the workstation:

```
ps -ef | grep <TWSuser>
```

This command list processes on the workstation and filters the output to the processes owned by the *<TWSuser>*. You should see a similar list of processes in Example 6-5.

Example 6-5 List of Tivoli Workload Scheduler processes on the workstation

```
tws  381114       1   0   Apr 04      -  0:28 netman
     tws  417832  438406  12 07:25:38  pts/0  0:00 ps -ef
     tws  438406 1339516   1 06:18:04  pts/0  0:00 -ksh
     tws  503886  438406   1 07:25:38  pts/0  0:00 grep tws
     tws 1179678  381114   0 06:22:49      -  0:00
/usr/tivoli/tws/bin/mailman -parm 320
     tws 1388624 1179678   0 06:22:49      -  0:03
/usr/tivoli/tws/bin/batchman -parm 320
     tws 1396902  381114   0 07:15:22      -  0:00
/usr/tivoli/tws/bin/writer -- 2001
    root 1405014 1388624   0 06:22:49      -  0:00
/usr/tivoli/tws/bin/jobman
```

Checking status locally on Microsoft Windows workstations

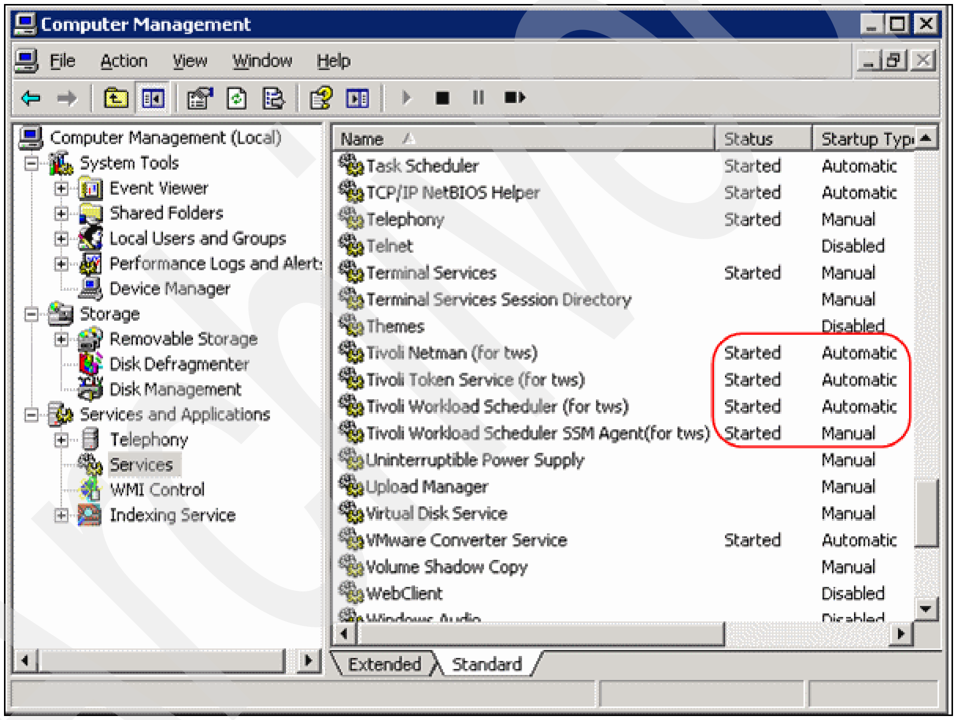
This section describes how to check whether the Tivoli Workload Scheduler services and processes on the Windows workstation are running.

Checking running services

Perform the following check:

1. Open the Windows Service Control Manager.
2. Check whether Tivoli Workload Scheduler services are in the Started status.

Figure 6-7 on page 187 shows the status of Tivoli Workload Scheduler services on the Windows workstation.



- Computer Management (Local)
  - System Tools
    - Event Viewer
    - Shared Folders
    - Local Users and Groups
    - Performance Logs and Alerts
    - Device Manager
  - Storage
    - Removable Storage
    - Disk Defragmenter
    - Disk Management
  - Services and Applications
    - Telephony
    - Services
    - WMI Control
    - Indexing Service

Name	Status	Startup Type
Task Scheduler	Started	Automatic
TCP/IP NetBIOS Helper	Started	Automatic
Telephony	Started	Manual
Telnet		Disabled
Terminal Services	Started	Manual
Terminal Services Session Directory		Manual
Themes		Disabled
Tivoli Netman (for tws)	Started	Automatic
Tivoli Token Service (for tws)	Started	Automatic
Tivoli Workload Scheduler (for tws)	Started	Automatic
Tivoli Workload Scheduler SSM Agent(for tws)	Started	Manual
Uninterruptible Power Supply		Manual
Upload Manager		Manual
Virtual Disk Service		Manual
VMware Converter Service	Started	Automatic
Volume Shadow Copy		Manual
WebClient		Disabled
Windows Audio		Disabled

Figure 6-9 Checking status of services on Microsoft Windows workstations

Checking running processes

This section describes how to check whether the Tivoli Workload Scheduler processes are running on a Microsoft Windows workstation. Two methods enable you to check running processes:

- Using the Task Manager: Open the Task Manager and then check whether processes listed in the section "Microsoft Windows platform" on page 185 are running.
- Using the utility listproc provided with Tivoli Workload Scheduler: Open the command-line interface and go to <*TWS_HOME*>/unsupported. Issue the following command:

  `listproc`

  You view the list of running processes on the system. Check whether Tivoli Workload Scheduler processes are in the output. Example 6-6 shows the output of the **`listproc`** command.

Example 6-6 Output of the listproc command

```
C:\Program Files\IBM\TWS\tws\unsupported>listproc | more
PID  Command                # Handles   # Threads
         0  Idle                        0           2
         4  System                   3112          76
       624  monman                    104           3
      1396  ssmagent                  271          21
      1556  taskmgr                    80           3
      3208  mmc                       225           3
       744  mmc                       304           6
      4768  cmd                        52           1
      1256  wuauclt                   130           4
      4424  tokensrv                   69           8
      5468  netman                     79           5
      4288  notepad                    47           1
       924  mailman                    93           2
      4572  batchman                   89           2
      5060  batchup                    62           4
      5260  JOBMAN                     93           2
      3664  JOBMON                    104           3
      6112  notepad                    61           2
```

Checking Tivoli Workload Scheduler status

This section describes how to check that the Tivoli Workload Scheduler is correctly running.

Issue the following command locally on the workstation:

`conman status`

You should receive a `Batchman LIVES` message as shown in Example 6-7.

Example 6-7 Checking Tivoli Workload Scheduler status

```
svcz0etepc08:/home/tws> conman status
Tivoli Workload Scheduler (UNIX)/CONMAN 8.4 (20070922) Licensed Materi
5698-WSH
(C) Copyright IBM Corp 1998, 2007 All rights reserved.
US Government User Restricted Rights
Use, duplication or disclosure restricted by GSA ADP Schedule Contract
IBM is a registered trademark of International Business Machines
Corporation in the United States, other countries, or both.
Installed for user "tws".
Locale LANG set to the following: "en"
Scheduled for (Exp) 05/14/08 (#1070) on SVCZ0ETEPC08.  Batchman LIVES.
Tivoli Workload Scheduler (UNIX)/CONMAN 8.4 (20070922)
```

Checking proper flags in showcpus

This section describes how to check whether the proper flags are set on the workstation.

Issue following command from the workstation's domain manager:

`conman showcpus cpu=<workstation>`

You should receive output similar to that shown in Example 6-8.

Example 6-8 Checking the workstation state from its domain manager

```
%sc cpu=svcz0eteph09
CPUID              RUN NODE        LIMIT FENCE DATE     TIME  STATE
SVCZ0ETEPH09      1070  UNIX FTA      60     0 05/14/08 00:03  LTI JW
MD
```

Important flags for the State field indicate the following:

- L - Workstation is linked to its domain manager.
- T - Workstation is linked directly over TCP/IP.

- I - jobman completed startup initialization.
- J - jobman is running.
- W - Writer is active.
- M - Monitoring agent has started.

Checking network communication ports

This section describes how to check whether the workstation's process netman listens on its default port and whether the bidirectional connection between the workstation and its domain manager has been established.

Issue the following command locally on the workstation:

`netstat -a`

You should receive output similar to that shown in Example 6-9.

Example 6-9 Checking the network communication ports

```
svcz0eteph09:/home/qrlisymar# netstat -a |grep 3111
tcp        0      0  *.31111                *.*
LISTEN
tcp4       0      0  svcz0eteph09.cez.31111 svcz0etepc08.cez.33905
ESTABLISHED
tcp4       0      0  svcz0eteph09.cez.36604 svcz0etepc08.cez.31111
ESTABLISHED
```

In Example 6-9, the ports in the left column are local ports; the ports in the right column belong to the remote systems. Therefore by reading the output of **`netstat -a`** you can determine the following:

- **`netman`** is listening on its default port.
- A bidirectional connection between the workstation and its domain manager has been established.

Note: You can see that the connection has been established, which does not mean that the workstations currently communicate actively.

6.7.3 Stopping workstations in Tivoli Workload Scheduler network

This section describes how to issue the `stop` command to Tivoli Workload Scheduler workstations across the domain structure. We discuss following topics:

- "Hierarchical stop using conman stop command" on page 196
- "Stop in firewall environments using conman stop;progressive command" on page 197

Hierarchical stop using conman stop command

This section describes the use of the `conman stop` command. The `conman stop` command can be used to stop the following:

- A single workstation
- Multiple workstations in one domain
- One or more workstations in more than one domain
- All workstations in the whole Tivoli Workload Scheduler network

Note: When you issue the `conman stop` command, you must have the "stop" privilege to workstations that you want to stop. You assign this privilege in the Security file.

The basic syntax of the command is:

```
conman stop [domain!]workstation
```

If you do not specify the domain, the command acts in the current domain (that is, the domain of the workstation where the command is issued).

Table 6-1 shows the rules and restrictions of the `conman stop` command when issued from different types of workstations.

Table 6-1 Rules and restrictions of stop command based on source

Workstation type where conman stop is issued	Rule and restriction
Master domain manager	Can stop any workstation in the Tivoli Workload Scheduler network
Domain manager	▸ Can stop any workstation in current domain and in the subordinate domains ▸ Cannot stop workstations in peer domains
Agent	Can stop any workstation in current domain

Stop in firewall environments using conman stop;progressive command

This section describes the use of the **`conman stop;progressive`** command. This command stops workstations hierarchically. It is similar to the **`conman stop @!@`** command (which stops all workstations in all domains), but it is more efficient in firewall environments because each domain manager by itself stops its own workstations. Use this command when you have defined at least one workstation as BEHINDFIREWALL.

When you issue the command on a domain manager, all workstations in that domain are stopped, and then the domain manager itself is stopped. The command continues to run in any subordinate domain. If any additional subordinate domains exist, the same steps are performed recursively.

Figure 6-10 describes the sample Tivoli Workload Scheduler network divided into four domains.

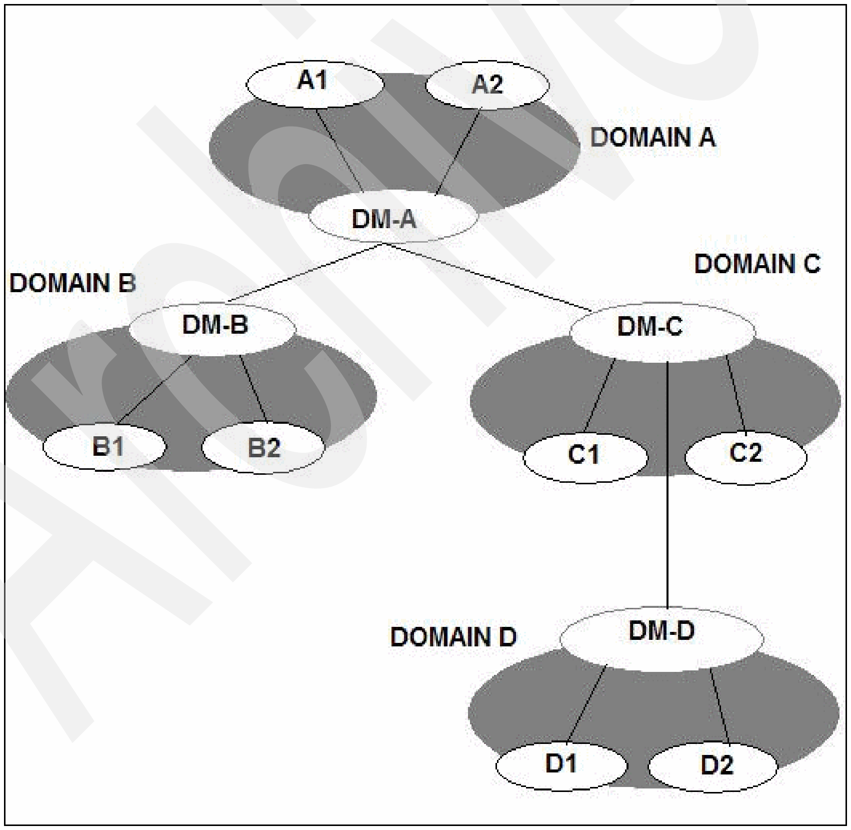


Figure 6-10 Sample hierarchical network

Example 6-10 on page 203, we issue the command **`conman stop;progressive`** on the workstation DM-C, which is the domain manager of the DOMAIN C. The following workstations are stopped in this order:

- In DOMAIN C (by the domain manager DM-C):
  - C1,C2
  - DM-C
- In Domain D (by the domain manager DM-D):
  - D1, D2
  - DM-D

6.7.4 Maintaining a workstation's file systems

This section describes how to maintain files and directories in the Tivoli Workload Scheduler installation directory. You must perform this task regularly to avoid the file system from becoming full. We focus on the following topics:

- "Location of log and trace files" on page 199
- "Location of joblog files" on page 199
- "Automatic housekeeping of logs, traces, and joblogs" on page 199
- "Maintaining archived, forecast, and trial plan files" on page 200
- "Maintaining audit files" on page 201

All the files and directories described in the following sections must be maintained manually regardless of whether a Tivoli Workload Scheduler cleanup utility is provided.

By default, Tivoli Workload Scheduler performs no automatic housekeeping of the files and directories listed in the following sections. The best way to perform regular housekeeping is to define jobs and job streams that run periodically and perform the cleanup of older logs, traces, joblogs, audit files, and so on.

We describe particular housekeeping methods in the following sections.

Location of log and trace files

This section describes the locations of log and trace files. These files are created on all workstations in the Tivoli Workload Scheduler network.

Table 6-2 describes the locations of log and trace files created by Tivoli Workload Scheduler processes. We also provide the maintenance method that you use for regular housekeeping of these files.

Table 6-2 Log and trace files of Tivoli Workload Scheduler processes

File type	Directory	Maintenance method
Log files	*<TWS_HOME>*/stdlist/logs	rmstdlist
Trace files	*<TWS_HOME>*/stdlist/traces	rmstdlist

By default all Tivoli Workload Scheduler processes except netman write into one common file. If you want to create a separate file for each process, you must override the default setting by specifying the following local option in the *<TWS_HOME>*/localopts file:

```
merge stdlist = no
```

Note: netman always logs on to the separate file even if you have the local option `merge stdlist` set to yes.

Location of joblog files

This section describes location of joblog files. Each Tivoli Workload Scheduler job creates a file containing the output of the script (or command) that has been executed. These files are called *joblogs* and are stored locally on the workstation where the job has run.

The following list describes the locations of joblog files and the maintenance method that is used for regular housekeeping of these files:

- File type: joblog files
- Directory: *<TWS_HOME>*/stdlist/logs/*<date>*
- Maintenance method: rmstdlist

Automatic housekeeping of logs, traces, and joblogs

This section describes how to perform automatic cleanup of logs, traces, and joblogs. We describe the rmstdlist utility. By default, Tivoli Workload Scheduler does not perform automatic housekeeping of these files and directories.

The easiest way to automate housekeeping of logs, traces, and joblogs is to define a job in Tivoli Workload Scheduler that runs the **rmstdlist** command with the appropriate argument. Then define a job stream with a proper run cycle to run the job periodically.

For instance, if you want to remove all standard list files (and their directories) that are more than seven days old, perform the following steps:

1. Define a job that runs the following command:

   ```
   rmstdlist 7
   ```

2. Define a job stream containing this job. Specify a daily run cycle for this job stream. Select Everyday to run this job stream each day.

> **Important:** Be aware that **rmstdlist** deletes logs, traces, *and* joblog files. You cannot specify that you want to delete, for instance, only trace files and you want to keep joblog files.

Maintaining archived, forecast, and trial plan files

This section describes how to maintain the following files:

- Archived production plan files
- Forecast plan files
- Trial plan files

In this section we *do not* describe files related to current production (such as message files).

Table 6-3 describes locations of archived plan files, trial plan files, and forecast plan files. We also provide the maintenance method that should be used for regular housekeeping of these files. The files and directories listed in Table 6-3 are located on the master domain manager or backup master domain manager.

Table 6-3 Joblog files, forecast, and trial plans

File type	**Directory**	**Maintenance method**
Archived plans	*<TWS_HOME>*/schedlog	Manual
Trial plans	*<TWS_HOME>*/schedTrial	Manual
Forecast plans	*<TWS_HOME>*/schedForecast	Manual

Tivoli Workload Scheduler does not provide a utility that performs cleanup of these files and directories. You must clean them up manually or develop a script that deletes unnecessary plan files. The latter method is usually date-based for archived plans and custom for forecast and trial plans.

To automate the cleanup process, define Tivoli Workload Scheduler jobs that call custom scripts. Then define job streams that contain these housekeeping jobs. You must define proper run cycles for these job stream as well.

Important: Archived plans, forecast plans, and trial plans are located only on the master domain manager (and eventually on the backup master domain manager). They are not located on domain managers or fault-tolerant agents.

Maintaining audit files

This section describes how to maintain files and directories that are used by the audit feature. Tivoli Workload Scheduler uses the audit feature to track operators' changes to the database and the plan.

Table 6-4 describes locations of archived plan files, trial plan files, and forecast plan files. We also provide the maintenance method that should be used for regular housekeeping of these files. These files and directories listed in Table 6-4 are located on the master domain manager or backup master domain manager.

Table 6-4 Directories used by the audit feature

File type	Directory	Maintenance method
Database audit files	<*TWS_HOME*>/audit/database	Manual
Plan audit files	<*TWS_HOME*>/audit/plan	Manual

The audit feature is disabled by default. You can turn on the audit feature by modifying the global options. The changes are effective as follows:

- Database audit - After restarting the master domain manager
- Plan audit - After the production plan extension

Table 6-5 lists the global options affecting the audit feature behavior.

Table 6-5 Global options related to the audit feature

Audit type	Global option	Value for turning on	Default value (meaning)
Database	enPlanAudit	1	0 (database audit off)
Plan	enDbAudit	1	0 (plan audit off)

Note: Even if by default the audit feature is off, the changes made to the security file are always logged as database changes. Every time you issue the **makesec** command, audit messages are written to files in the *<TWS_HOME>*/audit/database directory.

You therefore need to periodically maintain the *<TWS_HOME>*/audit/ directory even if you do not turn on the audit feature.

Tivoli Workload Scheduler does not provide a utility that performs cleanup of these files and directories. You must clean them up manually or develop a script that deletes unnecessary audit files. The script is usually date-based.

To automate cleanup, you should define Tivoli Workload Scheduler jobs that call your custom scripts. Then you must define job streams that contain these housekeeping jobs. You must define proper run cycles for these job streams as well.

Important: Database audit files are located only on the master domain manager (and eventually on the backup master domain manager). They are not located on domain managers or fault-tolerant agents.

Plan audit files are on the local workstation if the plan is accessed locally and the plan audit is active.

6.7.5 Determining Tivoli Workload Scheduler version and patch level

This section describes how to determine the current version and patch level of a Tivoli Workload Scheduler workstation. Information about the current version and patch level is stored in the *<TWS_HOME>*/version directory.

The name of the file containing the essential information is different on the Microsoft Windows and UNIX and Linux platforms. The following file names apply according to the platform:

- Microsoft Windows platform - maestro.info
- UNIX and Linux platforms - version.info

Example 6-10 shows an excerpt of the maestro.info file on the Microsoft Windows platform. The content of the version.info on the UNIX and Linux platform is similar.

Example 6-10 Content of maestro.info on the Microsoft Windows platform

```
## IBM Tivoli Workload Scheduler 8.4 Windows w32-ix86
## (C) Copyright IBM Corp. 1998, 2007
## PATCH 8.4.0-TIV-TWS-FP0001

#FILE                                        REVISION  PATCH
CHECKSUM     SIZE(BYTES)
bin/appservman.exe                          8.4      20080317
-----          131072
bin/archiver.exe                            8.4      20080317
-----           65536
bin/batchman.exe                            8.4      20080317
-----          389120
bin/batchup.exe                             8.4      20080317
-----           86016
```

The header of the file includes the version, hosting platform, and patch level. The rest of the file contains detailed information about each important binary file.

Note: Both version and patch level are included in the same file in the same directory - <*TWS_HOME*>/version/maestro.

6.8 Resetting and scratching the production plan

This section describes how to reset or scratch the current production plan and subsequent administrative tasks. The production plan can be scratched only on the currently acting master domain manager.

The difference between resetting and scratching the production plan is the following:

- If you *reset* the production plan, the production plan is deleted, and the preproduction plan is kept. The preproduction plan is updated with job statistics, and it is used later to generate a new production plan. This means that when you create a new production plan, it contains all job stream instances that were not in the COMPLETE state when you ran the **`ResetPlan`** command.

The plan can be reset by issuing the **`ResetPlan`** command without any arguments. The following steps are performed when you reset the production plan:

a. All Tivoli Workload Scheduler processes are stopped on the master domain manager.

b. The current Symphony file is archived.

c. The job statistics are updated.

- If you *scratch* the production plan, both the production plan and preproduction plan are scratched. The preproduction plan is created again based on the modeling information stored in the database. The new production plan contains all job stream instances scheduled to run in the current production period *regardless* of whether they were already in the COMPLETE state.

  The plan can be scratched by issuing the **`ResetPlan -scratch`** command. The following steps are performed if you scratch the production plan:

  a. All Tivoli Workload Scheduler processes are stopped on the master domain manager.

  b. The current Symphony file is archived.

  c. The job statistics are updated.

  d. The preproduction plan is scratched.

Important: If you have scratched the production plan and you are using DB2 as your persistent data storage, do not forget to run the dbrunstats utility. Execute this utility when all processes on the master domain manager are stopped (including the WebSphere Application Server). See 6.5, “Administering the DB2 database” on page 172 for details.

After resetting or scratching the production plan, you do not have a Symphony file on your master domain manager. When issuing various **`conman`** commands, you receive an error message indicating that the Symphony file is missing.

For instance, when you issue the **`conman status`** command after you have scratched the production plan, you receive a message similar to the one shown in Example 6-11.

Example 6-11 conman output when Symphony file is missing

```
C:\>conman status
Tivoli Workload Scheduler (Windows)/CONMAN 8.4 (20080317) Licensed
Materials - Property of IBM(R)
```

```
5698-WSH
(C) Copyright IBM Corp 1998, 2007 All rights reserved.
US Government User Restricted Rights
Use, duplication or disclosure restricted by GSA ADP Schedule Contract
with IBM Corp.
IBM is a registered trademark of International Business Machines
Corporation in the United States, other countries, or both.
Installed for user "BRUGE\tws".
Locale LANG set to the following: "en"
AWSBHU001E Conman encountered an error when attempting to open the
Symphony file: the file does not exist or conman could not
 find it. The following gives more details of the error: AWSBCU035E
Opening C:\Program Files\IBM\TWS\tws\Symphony, error: No
such file or directory.No Symphony file on workstation BRUGE, Audit
Level: 0
Tivoli Workload Scheduler (Windows)/CONMAN 8.4 (20080317)
No Symphony file on workstation BRUGE, Audit Level: 0
```

You must generate a new Symphony file (and therefore a new production plan) to recover from resetting (or scratching) the production plan. You can generate the new production plan by issuing the following command:

```
JnextPlan
```

This command generates a new production plan and distributes it to all workstations in the Tivoli Workload Scheduler network.

Important: Newly generated Symphony files have the limit for all workstations set to zero, which means that no jobs (except those with priority GO) can run. You must adjust the limit of the workstations to a higher value than zero to enable jobs to run. This step should be performed after you have made sure that job streams in the current production plan do not cause harm (for instance, you do not run the importing of credit card transactions for a second time). Bear in mind that resetting (or scratching) the production plan is a nonstandard situation, and you should carefully check which job streams are scheduled in the regenerated production plan.

7

Troubleshooting

This chapter provides troubleshooting information for the Tivoli Workload Scheduler V8.4 engine and its installation. In this chapter we discuss the following topics:

- “Installation troubleshooting” on page 208
- “Other common problems” on page 219
- “Engine log files - CCLog” on page 227
- “Embedded WebSphere Application Server troubleshooting” on page 228

7.1 Installation troubleshooting

In this section we describe issues dealing with the installation, removal, and configuration of Tivoli Workload Scheduler V8.4 and its prerequisites. We discuss the following topics:

- "Installation log files" on page 208
- "Recovering a failed installation" on page 210
- "Verifying the installation" on page 215
- "Problems installing DB2" on page 216
- "Other installation problems" on page 218

7.1.1 Installation log files

Log files of the installation processes have various naming conventions, depending on the installation method. This section describes the log files and what to include if you want to package them for support. The following topics are discussed:

- "InstallShield wizard installation and uninstallation log files" on page 208
- "TWSINST log files" on page 209
- "Software package block log files" on page 209
- "DB2 installation log files" on page 209
- "Installation log files for the embedded version of WebSphere Application Server V6.1" on page 210

InstallShield wizard installation and uninstallation log files

The InstallShield Multi Platform (ISMP) wizard log file are as follows:

- <*tempDir*>/tws84/twsismp.log - Trace file
- <*tempDir*>/tws84/summary.log - Log file

  where:

  <*tempDir*> is the user's temporary directory on the following platforms:

  – UNIX: /tmp
  – Microsoft Windows: The default value is C:\Documents and Settings\<*installing_user*>\Local Settings\Temp\

For multiple installations on the same workstation, the log header and footer indicate the user ID (<*TWS_user*>) for which the installation was performed.

Note: If you are running a silent installation and the response file you are using is not syntactically correct, the installation fails without producing a log.

TWSINST log files

The twsinst log file is as follows:

<*tempDir*>/tws84/twsinst_<*operating_system*>_<*TWS_user*>^8.4.0.00.log

where:

- <*tempDir*> is the user temporary directory on the following platforms:
  - UNIX: /tmp
  - Microsoft Windows: The default value is C:\Documents and Settings\<*installing_user*>\Local Settings\Temp\
- <*operating_system*> is the operating system.
- <*TWS_user*> is the user for which Tivoli Workload Scheduler was installed.

Software package block log files

The IBM Tivoli Configuration Manager software package block log files are as follows:

- <*tempDir*>/FP_TWS_<*operating_system*>_<*TWS_user*>^8.4.0.00.log (agent SPB log file)
- <*tempDir*>/TWS_LP_<*TWS_user*>^8.4.0.00.log (agent NLS SPB log file)

  where:

  - <*tempDir*>: The user temporary directory on the following platforms:
    - UNIX**:** /tmp
    - Microsoft Windows**:** The default value is C:\Documents and Settings\<*installing_user*>\Local Settings\Temp\
  - <*operating_system*>: The operating system
  - <*TWS_user*>: The user for which Tivoli Workload Scheduler was installed.

DB2 installation log files

The DB2 installation log file is the following:

- UNIX: /tmp/DB2Setup.log
- Microsoft Windows: C:\Documents and Settings\<*installing_user*>\My Documents\DB2LOG

Installation log files for the embedded version of WebSphere Application Server V6.1

The application server installation has no log. However, if you update the application server - for example, during the application of Tivoli Workload Scheduler fix packs - a log is created that provides information about the update. The log can be found in the directory <*TWS_home*>/appserver/logs/update, where you can find a directory that identifies the fix pack, which contains a file called updatelog.txt.

The log for the startup of the application server can be found in the directory <*TWS_home*>/appserver/profiles/twsprofile/logs/startServer.log.

Packaging log files for support

If a problem occurs with an installation that you cannot resolve, IBM Software Support may ask you to send all of the installation log files. Include the following files:

- All the files and subdirectories in the <*tempDir*>/tws84/ directory
- The software package block log files (if applicable)
- The DB2 installation log
- The installation log of the embedded version of the WebSphere Application Server V6.1

Attention: Do not remove, add, or modify files in the <*tempDir*>/tws84/ directory. To do so might cause an installation to fail or prevent the recovery of a failed installation.

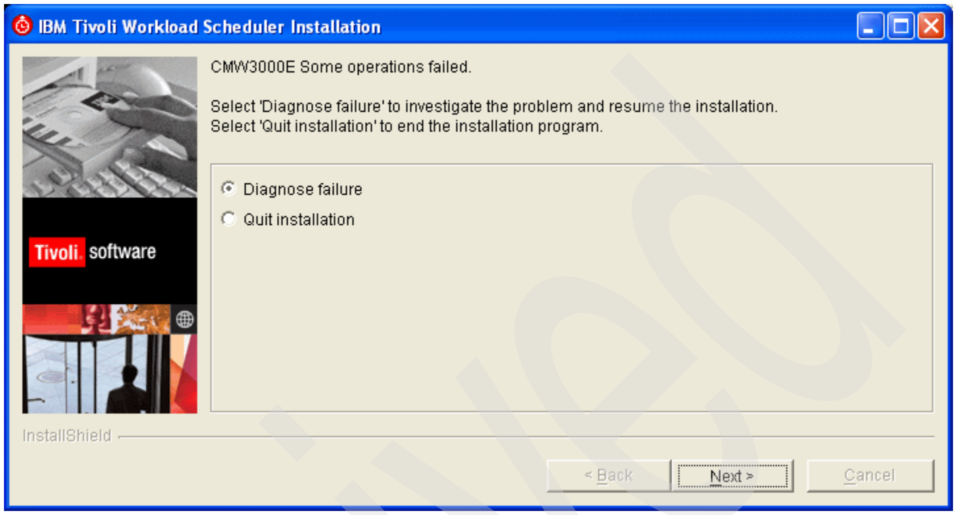
7.1.2 Recovering a failed installation

This section describes how to recover a failed installation of Tivoli Workload Scheduler V8.4. We discuss the following topics:

- "Recovering a failed installation with the InstallShield wizard" on page 211
- "Recovering a failed silent InstallShield wizard installation" on page 213

Recovering a failed installation with the InstallShield wizard

If an operation fails during the installation, the InstallShield wizard opens the CMW3000E Some operations failed panel (see Figure 7-1).



CMW3000E Some operations failed.

Select 'Diagnose failure' to investigate the problem and resume the installation.
Select 'Quit installation' to end the installation program.

[✓ Diagnose failure]
Quit installation

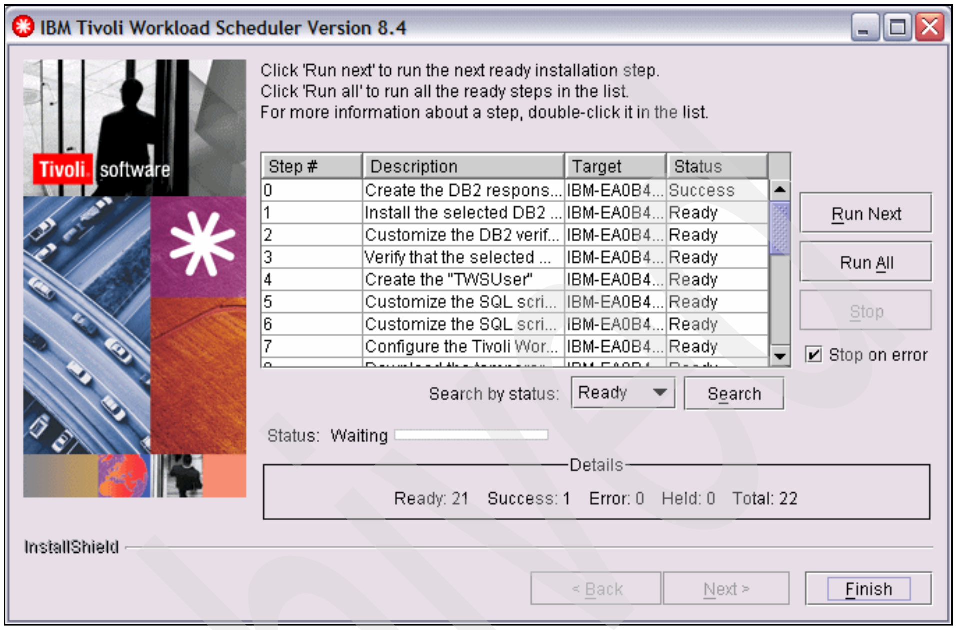
Figure 7-1 Installation failed panel

Attention: If you want to quit the installation, do not close the wizard by clicking on the **Close** icon. If you do, the wizard cannot save the troubleshooting information that you need for a resume. Instead select **Quit installation** and click **Next** to close the wizard.

Using the **Diagnose failure** option in the panel shown in Figure 7-1, you can see which step or steps of the installation have failed. You can correct errors that have occurred and resume those installation steps that have not completed successfully without leaving the wizard.

You can choose to perform these steps immediately after the failure occurs, or quit the installation and recover the situation later.

The Step List window is displayed either when an installation fails or when you resume an installation that previously has been stopped. Figure 7-2 shows the Step List window that is displayed when an installation step has failed.


IBM Tivoli Workload Scheduler Version 8.4

Click 'Run next' to run the next ready installation step.
Click 'Run all' to run all the ready steps in the list.
For more information about a step, double-click it in the list.

Step #	Description	Target	Status
0	Create the DB2 respons...	IBM-EA0B4...	Success
1	Install the selected DB2 ...	IBM-EA0B4...	Ready
2	Customize the DB2 verif...	IBM-EA0B4...	Ready
3	Verify that the selected ...	IBM-EA0B4...	Ready
4	Create the "TWSUser"	IBM-EA0B4...	Ready
5	Customize the SQL scri...	IBM-EA0B4...	Ready
6	Customize the SQL scri...	IBM-EA0B4...	Ready
7	Configure the Tivoli Wor...	IBM-EA0B4...	Ready

Search by status: Ready

Status: Waiting

Details

Ready: 21 Success: 1 Error: 0 Held: 0 Total: 22

Figure 7-2 Step List window

For each step shown in Figure 7-2, you can view the step number, description, target machine, and status. A step window opens when you double-click a step in the step list window. It contains three tabs:

- Status tab

  Displays the status of the installation step (Ready, Success, Error, or Held). You can change the status from Error to Ready if the condition that caused a step to fail has been removed.

- Properties tab

  Shows the user parameters required by the step. These might be the parameters you input in the installation wizard, or values that the wizard has determined according to the logic of its operations.

- Output tab

  Shows the output and any errors that occurred for the installation step, and also the commands that were performed by the installation.

You can stop and resume an installation at any time - for example:

- The installation is running successfully, but you want to pause it. You can stop it at this point and resume it later.
- The installation has failed, and you need to reboot the computer to correct a problem.

To stop an interactive installation that is running, click **Stop**. The wizard asks you if you want to quit the installation after the current step is completed. If you reply Yes, the installation completes the step being performed and then displays a summary panel of the completed activities. Click **Finish** to exit.

To stop an installation that has just failed, select **Quit installation**, click **Next**, and confirm your intention. To stop an installation that is on the Step List window, click **Cancel**, and confirm your intention.

To resume the installation, enter the following command:

```
<setup_file_name> -resume
```

where <*setup_file_name*> is one of the following:

- UNIX: setup.bin
- Microsoft Windows: setup.exe

Important: On Microsoft Windows, only a member of Administrators group can run SETUP.exe successfully.

When the InstallShield wizard recognizes that a previous installation has not completed, the Step List window opens. From here you can continue the previous installation from the point where it stopped. If the steps in the Step List window list no errors, you can resume the installation; otherwise you need to correct the error that caused the installation step (or steps) to fail before resuming that step (or steps).

Recovering a failed silent InstallShield wizard installation

If the silent wizard stops, follows this procedure:

1. Open the installation log and establish at what point the installation failed. The location of the installation log files is described in “Installation log files” on page 208.

   The installation is performed in two distinct phases:

   – Validation phase: The input parameters are validated, but no installation action is taken. In the log file, the validation of the input parameter is indicated by the action: validateFields.

   - Step execution phase: The installation is performed in a series of steps. In the log, each step starts with a message that begins “Running step”.
2. When you discover the nature of the problem and in what phase it started, you can then determine how to resolve it. You might have to correct a parameter or make a change in the installation environment (for example, create more disk space).
3. When you have resolved the problem, you can rerun or resume the wizard:
   - Rerun the wizard

     You must rerun the wizard if you find the error in the Validation phase.

     If the error occurred in the Step execution phase, you can always rerun it, but you must consider that the wizard attempts to redo each step. Thus, you might need to clean up the installation environment after the failed installation before rerunning it.

     If, for example, the DB2 installation was successful, rerunning the installation might cause the reinstallation of DB2 to fail. Thus, you must select the option in the response file to use the existing instance of DB2 that you have already installed.

     If you need to change an input parameter, edit the response file. Then rerun the wizard, reissuing the silent wizard command as you issued it originally.
   - Resume the wizard

     You can resume the wizard only if you find the error in the Step execution phase. The resume option uses the interactive wizard. You cannot resume the wizard silently because an interaction is required to resume the failed step.

     To resume the wizard, reissue the silent wizard command as you issued it originally but with these changes:

     - Add the parameter -resume
     - Remove the parameter -silent when you ran it originally. If you do not remove this parameter, the installation cannot resume.

     The Step list window of the interactive wizard is displayed, where you can optionally change the values of the data input parameters and resume the installation at the failed step.

Important: Full details of the steps are given in “Resolving installation problems, step-by-step” in Chapter 10 of the *IBM Tivoli Workload Scheduler Administration and Troubleshooting V8.4* (revised March 2008), SC32-1275.

7.1.3 Verifying the installation

After installing the product, the installation process proceeds to complete the following configuration tasks:

- Create the main local and global default settings.
- Configure security access.
- Creates a default Security file in the <*TWS_home*> directory. By default, it authorizes <*TWS_user*> and the administrator or root user. It is also updated when a full InstallShield wizard installation of the connector is performed.
- Set workstation and user definitions each time you install or promote a Tivoli Workload Scheduler master domain manager.

To make sure no errors occurred during installation, check the install log file (see 7.1.1, “Installation log files” on page 208 for information about the log files and where to find them).

The following examples are checks you can perform to verify the installation, and the corresponding recovery actions:

- Check the main local and global default settings.

  If you promoted a workstation from the role of standard agent or fault-tolerant agent to the role of master domain manager, check that the master global option is set to the correct workstation name.

  If it is wrong, you must manually edit the files to replace the current values with the correct ones.
- Check for the Security file.

  Check that the Security file, the default operational security file, was created in the <*TWS_home*> directory. If it was not, create the file as follows:

  a. Set the Tivoli Workload Scheduler environment by running the script tws_env.
  b. Customize the Security file, as follows:
     i. Open the file <*TWS_home*>/config/Security.conf.
     ii. Edit the contents to reflect your environment and requirements.
     iii. Save the file as <*TWS_home*>/Security.conf.
- Run one of the following commands:
  – UNIX: `makesec -l Security.conf`
  – Microsoft Windows: `makesec Security.conf`

- Check for workstation and user definitions.

  Check that the required workstation and user definitions are in place in the database of the master domain manager. To add missing definitions in the database, follow the instructions in the *IBM Tivoli Workload Scheduler IBM Tivoli Workload Scheduler Job Scheduling Console User's Guide V8.4*, SC32-1257, or *IBM Tivoli Workload Scheduler Reference Guide V8.4* (revised March 2008), SC32-1274.

7.1.4 Problems installing DB2

The installation of DB2 is performed silently, driven by a response file generated by the Tivoli Workload Scheduler InstallShield wizard. It might encounter any one of a number of problems:

- The installation fails because the prerequisites for installing DB2 with an InstallShield wizard are not met (for example, kernel patches).
- The installation fails because the prerequisites for installing DB2 are not met (for example, kernel patches).
- The installation succeeds but is not customized for your environment because the wizard uses default values for certain DB2 installation parameters that do not meet the requirements of your environment.
- The installation succeeds, but DB2 cannot start because the prerequisites for running DB2 are not met (for example, kernel patches and parameters).

Two potential solutions can solve this problem:

- You can install the supplied version of DB2 separately, before installing Tivoli Workload Scheduler. Use the DB2 installation documentation to determine how to define the environment you require. You then continue the installation of Tivoli Workload Scheduler, referring to the freshly installed instance of DB2 as the existing instance you want to use for Tivoli Workload Scheduler. This approach is in accordance with the license for using DB2 with Tivoli Workload Scheduler.
- You can refine the UNIX kernel parameters using DB2 utilities, such as **`db2osconf`**. This tool is installed with DB2 and can be found in the binaries directory:
  - AIX: /usr/opt/db2_08_01
  - Other UNIX: /opt/IBM/db2/V8.1

It is used after installation to refine the kernel parameters.

In addition to the previously listed problems, the following specific problems might occur when installing DB2:

- The DB2 client fails; refer to “The installation of the DB2 client fails” on page 217.
- Creation of the DB2 tablespace fails; refer to “The creation of the DB2 tablespace fails” on page 217.

The installation of the DB2 client fails

You have chosen to install the DB2 client as part of your Tivoli Workload Scheduler installation, and the installation has failed.

Cause and solution

Several cause are responsible for this problem, but the most likely is that when performing the installation, you did not have write permission for the chosen installation directory, or if you used the defaults to the home directory of the <*TWS_user*>.

Do one of the following:

- Add the appropriate permissions and resume the installation at the point where it failed.
- Use a different user who has the appropriate permissions. In this case, you must rerun the installation because you cannot resume an installation if the user has changed.

The creation of the DB2 tablespace fails

You are installing DB2 or using an existing instance but have chosen a directory outside the DB2 installation structure for the tablespace. The installation fails in the Configure the Tivoli Workload Scheduler database step, and in the DB2 installation log you find the following message:

```
DB21034E The command was processed as an SQL statement because it was
not a valid Command Line Processor command. During SQL processing it
returned: SQL0970N The system attempted to write to a read-only file.
SQLSTATE=55009
```

Cause and solution

One possible reason for the problem is that the directory in which you created the tablespace directory did not have write permission.

To resolve the problem, perform the following steps:

1. Give write permission to all users for the parent directory of the tablespace directory
2. Resume the installation at the failed step.

7.1.5 Other installation problems

The following miscellaneous problems can occur:

- An installation fails on a UNC mapped drive.
- You receive the message “Error writing file=received”.

See the following sections for information.

Installation fails on a UNC mapped drive

You are running an installation with the installation images on a drive mapped using the Universal Naming Convention (UNC). The wizard fails at the first step.

Cause and solution

The Tivoli Workload Scheduler installation wizard methodology does not support UNC mapped drives. Rerun the installation from a drive that is not UNC mapped.

Message “Error writing file =” received

When performing any type of installation on any operating system, you might receive the following error:

```
Error writing file = There may not be enough temporary disk space. Try
using -is:tempdir to use a temporary directory on a partition with more
disk space.
```

Cause and solution

This error means what it says; the solution is as follows.

First, try to redirect the installation to use a different temporary directory by adding the **`-is:tempdir`** `.`*`<temp_dir_path>`* variable to the installation command.

If this does not work, you must use one of these two methods to give more space to the swdis directory:

- Create a new version in a different file system. The procedure is as follows:
  a. Delete or rename both the work and the backup subdirectories and recreate the directories in a file system with more space in it.
  b. Link the new directories to the .swdis directory using the **`ln -s`** command.

- Create a new backup directory in a file system with more space in it, and modify the /etc/Tivoli/swdis.ini file to point to it. Be sure to modify the correct section of the swdis.ini file, as follows:
  - If you are making a local silent InstallShield wizard installation that uses the disconnected command line (**`wdinstsp`**), modify the value of the backup_dir key in the [#MOBILE] section.
  - If you are making a remote installation using Tivoli Configuration Manager, you must identify the section relative to the endpoint chosen as the target (for example, `[my_aix_machine]`), and modify the backup_dir key in that section.

7.2 Other common problems

This section discusses commonly occurring problems and their solutions. We discuss the following topics:

- "Problems on Microsoft Windows" on page 219
- "Symphony problems" on page 222
- "JnextPlan problems" on page 222
- "Production problems" on page 224
- "Communication problems" on page 226

7.2.1 Problems on Microsoft Windows

You may encounter the following problems running Tivoli Workload Scheduler on Microsoft Windows:

- The interactive job is not interactive using Terminal Services.
- Tivoli Workload Scheduler services fail to start after installation.
- Tivoli Workload Scheduler for user service (batchup) fails to start.

Refer to the sections that follow for information about these problems.

Interactive job is not interactive using Terminal Services

You want to run a job at a Microsoft Windows fault-tolerant agent, launching the job remotely from another workstation. You want to use Microsoft Windows Terminal Services to launch the job on the fault-tolerant agent, either with the Job Scheduling Console or from the command line. You set the `is interactive` flag to supply runtime data to the job, and indicate the application program that is to be run (for example, notepad.exe).

Although everything seems correct, when the job starts running, the application program window does not open on the Terminal Services window. An investigation of the fault-tolerant agent shows that the application program is running on the fault-tolerant agent, but Terminal Services is not showing you the window.

Cause and solution

The problem is a limitation of Terminal Services, and no known workaround exists. All interactive jobs must be run by a user at the fault-tolerant agent and cannot be run remotely using Terminal Services. Jobs that do not require user interaction are not impacted and can be run from Terminal Services without any problems.

The Tivoli Workload Scheduler services fail to start after installation

On Microsoft Windows, both the Tivoli Token service and the Tivoli Workload Scheduler for user service (batchup) fail to start for the first time (after a successful installation).

Cause and solution

During the installation, you selected the option to create the <*TWS_user*>, but an error was found with the password you supplied (for example, the supplied password did not satisfy the security policy on the workstation). You used the restart facility (see "Recovering a failed installation with the InstallShield wizard" on page 211) to recover the situation. On the recovery panel, you entered a different value for the password than the value entered originally. This value is valid so the program completed the installation.

However, during the completion of the installation, the <*TWS_user*> was created using the new password that you entered on the recovery panel, while the services were created using the original password value.

The reason for this problem is that when the installation wizard starts the installation steps, after having accepted the input of all of the installation variables, it creates each of the steps complete with the properties (variables) required to perform the step. If a step fails and the Step List window opens, the properties that are displayed for one step are quite separate from the properties for another step. Thus, if you changed the password for one step, you must also change it for all the other steps where it is used.

- To resolve this problem after the installation has completed, you must change the password for the services to the value that you entered on the recovery panel.

- If you become aware of this problem before the installation is complete, you can choose one of the following alternatives:
  - Let the installation change the password afterward, as described in previously.
  - Exit from the installation, uninstall whatever you have installed following the procedures described in the *IBM Tivoli Workload Scheduler Planning and Installation Guide v8.4* (revised March 2008), SC32-1273, and rerun the installation.

Tivoli Workload Scheduler for user service (batchup) fails to start

The Tivoli Workload Scheduler for <*TWS_user*> service (sometimes also called *batchup*) does not start when the other Tivoli Workload Scheduler processes (for example, mailman and batchman) start on workstations running Microsoft Windows 2000 and Windows 2003 Server. This problem occurs on a fault-tolerant agent, either after a **`conman start`** command or after a domain manager switch. The Tivoli Token service and netman services are unaffected.

This problem does not impact scheduling but can result in misleading status data.

Cause and solution

The problem is probably caused either because the <*TWS_user*> has changed password, or because the name of the service does not match that expected by Tivoli Workload Scheduler. The latter can be a problem because a change in the configuration of the workstation has impacted the name of the service.

To resolve the problem temporarily, start the service manually using the Windows Services panel (under Administrative Tools). The service starts and runs correctly. However, the problem could reoccur unless you correct the root cause.

Note: Check how to resolve the problem permanently in *IBM Tivoli Workload Scheduler Administration and Troubleshooting V8.4* (revised March 2008), SC32-1275.

7.2.2 Symphony problems

This section describes how to diagnose and fix a corruption of the Symphony file.

Symphony file corruption is a rare event, and you must verify a potential corruption before taking action. Common symptoms are the following:

- A specific message informing you that the Symphony file is corrupt.
- A shutdown of various processes (especially batchman) with error messages referring to problems with the Symphony file in the stdlist.

The normal reason for the corruption of the Symphony file is a full file system. This can be avoided by regularly monitoring the file system where Tivoli Workload Scheduler is installed.

Recovery procedure for Symphony file on master domain manager

The following procedure can also be used to recover a corrupt Symphony file. The procedure does not require the use of a backup master domain manager.

Follow these steps on the master domain manager:

1. Set the job "limit" to 0, using **`conman`**, the Tivoli Dynamic Workload Console, or the Job Scheduling Console. This prevents jobs from launching.
2. Shut down all Tivoli Workload Scheduler processes on the master domain manager.
3. Run **`ResetPlan`**. (The CPU limit automatically changes to 0.)
4. Run **`JnextPlan`**, setting the `-from` and `-to` parameters to cover the period for which outstanding jobs still exist. Only incomplete job stream instances can be included in the new Symphony file.
5. Check the created plan and ensure that you want to run all the instances it contains, deleting those that you do not want to run.
6. Reset the job "limit" to the previous value. The Symphony file is distributed, and production recommences.

7.2.3 JnextPlan problems

This section describes common problems occurring during the JnextPlan.

- JnextPlan fails to start.

  This error might be a symptom indicating that your Tivoli Workload Scheduler network requires additional tuning. The default size of the production message box files is 10 MB. You might want to increase the size according to the following criteria:

- The role (master domain manager, domain manager, or fault-tolerant agent) of the workstation in the network. Higher hierarchical roles need larger pobox files due to the larger number of events they must handle (because the total number of events that a workstation receives is proportional to the number of its connections). For a domain manager, the number of subdomains under its control also can make a difference.
- The average number of jobs in the plan.
- The I/O speed of the workstation (Tivoli Workload Scheduler is I/O dependent).

- FINAL or JnextPlan fails with a Java out-of-memory error.

  You receive the following messages from JnextPlan or FINAL Job Stream:

  ```
  AWSJCS011E An internal error has occurred. The error is the
  following: "java.lang.OutOfMemoryError".
  ```

  The problem is probably caused by the number of jobs that JnextPlan has to handle. The default Java heap size in the application server cannot handle more than about 40 000 jobs. If JnextPlan is handling this many jobs, you need to increase the Java heap size. See *IBM Tivoli Workload Scheduler Administration and Troubleshooting V8.4* (revised March 2008), SC32-1275, for a full description of how to do this.

- JnextPlan fails with message AWSJPL017E. You receive the following message from JnextPlan:

  ```
  AWSJPL017E The production plan cannot be created because a previous
  action on the production plan did not complete successfully. See the
  message help for more details.
  ```

  The problem might be caused by a JnextPlan being launched before the previous JnextPlan has run the **`SwitchPlan`** command.

  The situation might not resolve itself. To resolve it yourself, do the following:

  a. Reset the plan by issuing the command **`ResetPlan -scratch`**.
  b. If the reset of the plan shows that the database is locked, run the **`planman unlock`** command.

7.2.4 Production problems

This section describes common problems that occur in a normal production environment.

- All jobs in ready status

  A new workstation has been added in your production environment. The next Final scheduler complete successfully, but all the jobs for that workstation stay in ready state.

  After adding a new workstation to the environment, the limit for the new CPU is automatically set to 0. In this case, you need to change the limit for this CPU to release the jobs.

- All jobs fails for a specific user

  On Microsoft Windows workstations, all jobs executing as a specific user begin to fail.

  Cause and solution: This error usually indicates that the user password was changed. To solve the problem you need to update the <*TWS_user*> password. Use **`composer`** to update the password in the database and **`conman altpass`** to change the password in the plan to the same value as you update the password in the database.

- Error: AWSBIJ109E. Jobmon was unable to log a user on to the local computer. In the TWSMERGE log file you can see the following error message:

  ```
  JOBMON:+ AWSBIJ109E An internal error has occurred.Jobmon was unable
  to log a user on to the local computer.
  ```

  Cause and solution: The error occurs only on Microsoft Windows workstations. It is usually caused by an incorrect user rights setting for the streamlogon user. The Microsoft Windows user needs the following rights to work correctly:

  – Act as part of the operating system
  – Increase quotas
  – Log on as batch job
  – Log on as a service
  – Log on locally
  – Replace a process-level token
  – Impersonate a client after authentication

- Job troubleshooting

  The internal status for a job will be one of the following:

  – **FAILED**

    This status means that the job is launched but never runs the script name, docommand, or JSC script file and produces no job log. This usually occurs in these cases:

    - The user does not exist.
    - The script does not exist.
    - The script that should be run by a job is not executable.
    - The user does not have the permission to execute the script.

    The best file to check in these cases is TWSMERGE under <*tws_home*>/stdlist/logs/<*yyyymmdd*>_TWSMERGE.log.

    Note: If you are on a Microsoft Windows workstation, ensure that the logon user has the Log on as a batch job user right.

  – **ABEND**

    The job is launched but finished with a non-zero return code. The best way to troubleshoot the error is to check the job log, using JSC or under: *TWShome*/stdlist/yyyy.mm.dd/O<*jobnumber*>.<*time*>.

  – **FENCE**

    The FENCE status means that the job cannot run because the fence of the workstation is equal to or higher than the priority of the job.

  – **SUCC**

    The SUCC status means that the job finished successfully. The default return code used to confirm the success of the job is 0; in other cases the job is considered in ABEND. You can use two other methods to define a job status of SUCC even if the script finishes with an exit code other than 0:

    - User return code mapping.
    - Define the job as “Require Confirmation” and then confirm it as SUCC.

7.2.5 Communication problems

This section describes common problems between the master domain manager and fault-tolerant agents.

- Connectivity

  The Netman process on the FTA is up and running, but it is not possible to link the agent to the master domain manager.

  Cause and solution: This problem is usually caused by firewall rules or network problems. Check the communication between the master domain manager and fault-tolerant agent by running the following command from the master domain manager:

  ```
  telnet <node> <port>
  ```

  where:

  – *<node>*: The host name of the FTA.

  – *<port>*: The port number used by netman process. To display the port number of netman you can also run the following command:

  ```
  conman showcpu;link
  ```

  Running telnet you can see if the port is open for communication. To check whether the port is in LISTEN, you can also run the following command locally on the FTA:

  ```
  netstat -a | grep <port>
  ```

  You can use the following commands to display the port number in the production plan that the master domain manager is currently using to connect with a fault-tolerant agent:

  – `cpuinfo <FTA> port`

  – `conman "showcpu <FTA>;link"`

  You must open the netman port to solve some link problems.

  For other connectivity problem, you can also check the NETMAN log under <*TWS_home*>/stdlist/logs/<*yyyymmdd*>_NETMAN.log and the TWSMERGE log under <*TWS_home*>/stdlist/logs/<*yyyymmdd*>_TWSMERGE.log. These two files can help you troubleshoot connectivity problems.

  Important: For Tivoli Workload Scheduler to run correctly, a two-way connection must exist between the domain manager and fault-tolerant agent.

- Link problems

  The master domain manager is unable to link to a fault-tolerant agent to refresh the production plan on the FTA.

  Cause and Solution: This problem has two main causes:

  – The netman process is down.

    Start the netman process on the FTA.

  – One of the .msg files has reached the maximum size.

    Stop all processes, remove all .msg files, the Symphony file, and the Sinfonia file and run the StartUp script to start netman.

- Network considerations

  Netman is started by the **`StartUp`** script (command). The order of process creation is netman, mailman, batchman, and jobman. On standard agent workstations, batchman does not run. All processes, except jobman, run as the TWSuser. Jobman runs as root.

  When network activity begins, netman receives requests from remote mailman processes. Upon receiving a request, netman spawns a writer process and passes the connection to it. Writer receives the message and passes it to the local mailman. The writer processes (more than one on a domain manager may exist) are started by link requests and are stopped by unlink requests (or when the communicating mailman terminates).

  Domain managers, including the master domain manager, can communicate with a large number of agents and subordinate domain managers. For improved efficiency, you can define mailman servers on a domain manager to distribute the communications load (see the section that explains how to manage workstations in the database in the *IBM Tivoli Workload Scheduler Job Scheduling Console User's Guide V8.4*, SC32-1257).

  You can configure netman to listen on a specific port, changing the following option in the localopts file:

  ```
  nm port <port>
  ```

7.3 Engine log files - CCLog

CCLog is a logging engine that creates log files with a defined structure. It can be used to monitor many products from a variety of software suppliers. The configuration supplied with Tivoli Workload Scheduler uses it uniquely for Tivoli Workload Scheduler processes.

CCLog file locations

The log files CCLog produces are stored in different places, depending on the settings in the localopts file:

- merge stdlists = yes
  - <*TWS_home*>/stdlist/logs/<*yyyymmdd*>_NETMAN.log - The log file for netman
  - <*TWS_home*>/stdlist/logs/<*yyyymmdd*>_TWSMERGE.log - The log file for all other processes
- merge stdlists = no

  <*TWS_home*>/stdlist/logs/<*yyyymmdd*>_<*process_name*>.log where <*process_name*> is one of the following:

  - BATCHMAN
  - JOBMAN
  - JOBMON
  - MAILMAN
  - NETMAN
  - WRITER

Low-level traces and open source library messages that do not conform to the current standard Tivoli Workload Scheduler message format (for instance, some SSL stdout and stderror messages) are found in the <*TWS_home*>/stdlist/yyyy.mm.dd/<*TWS_user*> file.

7.4 Embedded WebSphere Application Server troubleshooting

In this section we describes common procedures for troubleshooting the embedded version of WebSphere Application Server V6.1.

7.4.1 Log files

The default location of the embedded WebSphere Application Server log files is the following:

```
<TWS_home>\appserver\profiles\twsprofile\logs\server1
```

The directory contains the standard error and standard output logs, and the stop and start server’s log.

7.4.2 Application server properties

This section describes a common procedure that we recommend you follow when using these utilities:

- changeDataSourceProperties
- changeHostProperties
- changeSecurityProperties

To avoid the risk of changing a configuration value inadvertently, you follow a procedure that creates a file containing the current properties, edit it to the values you require, and apply the changes. The steps are as follows:

1. Log on to the computer where Tivoli Workload Scheduler is installed as the following user:
   – UNIX: root
   – Microsoft Windows: Any user in the Administrators group
2. Access the directory <*TWS_home*>\wastools.
3. Stop the WebSphere Application Server using the **`conman stopappserver`** command.
4. From the same directory run the following script to create a file containing the current properties:
   – UNIX: **`show`**<*property_type*>**`Properties.sh`** > my_file_name
   – Microsoft Windows: **`show`**<*property_type*>**`Properties.bat`** > my_file_name

   where <*property_type*> is one of the following:

   – DataSource

   The DataSource property file contains information about the RDBMS in use. Only the WebSphere Application Server resources.xml are affected by this script. The full path of the file is:

   <*TWS_home*>/appeserver/profiles/twsprofiles/config/cells/DefaultNode/nodes/DefaultNode/servers/server1/resources.xml

   – Host

   The Host property file contains information about the WebSphere Application Server host name and ports. Only the WebSphere Application Server serverindex.xml file is affected by this script. The full path of this file is:

   `<TWS_home>/appeserver/profiles/twsprofiles/config/cells/DefaultNode/nodes/DefaultNode/serverindex.xml`

- Security

  The Security property file contains the security settings related to the use of Tivoli Workload Scheduler, including global security settings, local OS settings, LDAP settings, and SSL key store settings. Only the WebSphere Application Server security.xml are affected by this script. The full path of this file is:

  ```
  <TWS_home>/appeserver/profiles/twsprofiles/config/cells/DefaultNo
  de/security.xml
  ```

5. Edit my_file_name with a text editor. Check the start of the file. The command might have written a message from the application server (WASX7357I:) at the beginning of the file. Delete this message.
6. Change the value of the configuration parameters according to your requirements. You do not have to supply all of the parameters in the file.
7. Save the file my_file_name.
8. Run the script:
   - UNIX: **`change`**<*property_type*>**`Properties.sh`** my_file_name
   - Microsoft Windows: **`change`**<*property_type*>**`Properties.bat`** my_file_name

     where <*property_type*> is the same used in step 4 on page 229. The properties are updated, according to the rules given in the descriptions of each property type.
9. Start the WebSphere Application Server using the **`conman startappserver`** command.
10. Check that the change has been implemented in Tivoli Workload Scheduler.

7.4.3 Database connection

After changing some application server properties, you must check whether the database connection is correct. To do so, you can use the following two commands:

- **`opman ls`** - This command shows the global options from the database.
- **`planman showinfo`** - This command shows the plan information from the database.

7.4.4 Updating the embedded version of WebSphere V6.1 services

Use the **updateWasService.bat** script to update the following WebSphere Application Server data on a Microsoft Windows operating system:

- The user ID and password of the local operating system user that runs the service.
- The user ID (wasuser) and password (waspassword) of the user required to stop the WebSphere Application Server, according to the user registry used by the server. If these are not provided, the operating system service credentials are used.
- An alternative location for the embedded version of WebSphere Application Server installation directory and profile.

  At run time, the script calls WASService.exe.

You can use the following syntax:

updateWasService.bat -userid <*TWS_user*> **-password** <*TWS_user_password*>

- **[-wasuser** <*WAS_user*> **-waspassword** <*WAS_user_password*>**]**
- **[-starttype {automatic | manual | disabled}]**
- **[wasHome** <*WebSphere_install_directory*>**]**
- **[profilePath <server_profile_directory]**

Change trace properties

The application server handles all communications between the Tivoli Workload Scheduler processes. The trace for these communications is set to None by default. The application server can be set to trace all communications, either for the whole product or for these specific groups of processes:

- Database
- Planner
- Command line
- Utilities
- Connector

Attention: Activating traces for the embedded version of the WebSphere Application Server V6.1 significantly impacts performance, especially if you set the tracing to All. We strongly recommend you identify the process group where the problem that you want to trace is occurring, and only set the trace to that group.

The procedure to set a trace is as follows:

1. Log on to the computer where Tivoli Workload Scheduler is installed as the following user:
   - UNIX: root
   - Microsoft Windows: Any user in the Administrators group
2. Access the directory <*TWS_home*>/wastools.
3. Stop and restart the application server.
4. Run the script:
   - UNIX: **`changeTraceProperties.sh-user`** <*TWS_user*>

     `-password` <*TWS_user_password*>

     `-mode trace_mode`
   - Microsoft Windows: **`changeTraceProperties.bat-user`** <*TWS_user*>

     `-password` <*TWS_user_password*>

     `-mode trace_mode`

   where the command attributes are as follows:

   - **`-user`** <*TWS_user*>

     The <*TWS_user*> ID
   - **`-password`** <*TWS_user_password*>

     The <*TWS_user*> password
   - **`-mode trace_mode`**

     Where trace_mode is one of the following:

     - **active_correlation:** All communications involving the event correlator are traced.
     - **tws_all_jni:** All communications involving JNI™ code are traced. ("JNI" stands for Java Native Interface.)
     - **tws_all:** All Tivoli Workload Scheduler communications are traced.
     - **tws_alldefault:** Resets the trace level to the default level imposed at installation.

- **tws_cli:** All Tivoli Workload Scheduler command-line communications are traced.
- **tws_conn:** All Tivoli Workload Scheduler connector communications are traced.
- **tws_db:** All Tivoli Workload Scheduler database communications are traced.
- **tws_planner:** All Tivoli Workload Scheduler planner communications are traced.
- **tws_secjni:** All Tivoli Workload Scheduler JNI code auditing and security communications are traced.
- **tws_utils:** All Tivoli Workload Scheduler utility communications are traced.

To reset the trace to null, either run the preceding procedure with trace_mode as tws_info, or just stop and restart the application server.

7.5 Reorganizing the DB2 database

The DB2 database has been set up to maintain itself, so little user maintenance is required. Periodically, DB2 checks the database by running an internal routine. DB2 determines when this routine must be run using a default policy. This policy can be modified, if need be, or can be switched off so that DB2 does not perform internal automatic maintenance. Using the statistical information that DB2 discovers by running this routine, it adjusts its internal processing parameters to maximize its performance.

This routine has also been made available for you to run manually in case you feel that the performance of DB2 has degraded, or because you have just added a large amount of data and anticipate performance problems. The routine is imbedded in the dbrunstats tool, which can be run to improve performance while DB2 is processing data, without causing any interruption.

It is also possible to physically and logically reorganize the database using the dbreorg script. This effectively recreates the tablespace using its internal algorithms to determine the best way to physically and logically organize the tables and indexes on disk. This process is time consuming and requires that Tivoli Workload Scheduler is down while it is run, but it does provide a freshly reorganized database after major changes.

dbreorg tool

Using this tool, the database physically reorganizes the data tables and indexes, optimizing disk space usage and ease of data access. The process is time consuming, requires that the database is backed up, and requires that Tivoli Workload Scheduler is stopped. However, when the tool completes, you have a database that is completely reorganized.

To reorganize the database follow this procedure:

1. Back up the Tivoli Workload Scheduler database.
2. Stop all Tivoli Workload Scheduler processes.
3. Check that the user who is going to run the procedure has the appropriate rights.
4. Open a DB2 shell.
5. Check that the command shell is correctly initialized by issuing the command `db2`, and checking that the command is recognized.
6. Issue the command `quit` to leave the DB2 Processor mode.
7. From within the shell, change to the directory <*TWS_home*>/dbtools/db2/scripts.
8. Run the script:
   - UNIX: **`dbreorg.sh`** *database* [*user* [*password*]]
   - Microsoft Windows: **`dbreorg`** *database* [*user* [*password*]]

     where:

     - *database*: The name of the database.
     - *user*: The DB2 administration user. If this is omitted, the ID of the user running the command is used.
     - *password*: The password of the DB2 administration user. If this is omitted, it is requested interactively.

   The script runs, issuing various messages denoting its progress and successful conclusion.
9. Restart Tivoli Workload Scheduler.

A

Sample certification test questions

In this appendix, we provide sample questions that are representative of the ones you encounter on the Tivoli Workload Scheduler Certification test. We recommend you take this sample test after studying the chapters in this book.

Questions

We provide the following questions to assist you in studying for the Certification test:

1. What is the full path name of the installation log of DB2 installed with the Tivoli Workload Scheduler master domain manager (MDM) on a UNIX system?
   a. /var/log/DBLog.txt
   b. /tmp/tws84/DB2/DB2LOG
   c. /opt/Tivoli/DB2/logs/install.log
   d. /tmp/DB2Setup.log
2. What command is used to see the global options on the master domain manager?
   a. **conman showopt**
   b. **composer display=options**
   c. **optman ls**
   d. planman -options
3. Which of the following configurations do you to change to modify the netman port? (Choose two.)
   a. globalopts on FTA
   b. localopts on FTA
   c. cpu/workstation definition
   d. planman showinfo
   e. /usr/tivoli/logs/MAESTRO.log
4. Which is the log file that contains information about all Tivoli Workload Scheduler processes excluding netman?
   a. *TWShome*/stdlist/logs/<*yyyymmdd*>_TWSMERGE.log
   b. /tmp/<*yyyymmdd*>_MERGELOG
   c. /tmp/tws84/logs/0<*time*>_tws.log
   d. 2000
5. What two commands can be used to reset the trace level to null in the embedded version of WebSphere Application Server V6.1?
   a. changeTraceProperties.sh(.bat) -user <*tws_user*> -password <*tws_user_password*> -trace_mode tws_info
   b. resetTraceProperties.sh(.bat) -user <*tws_user*> -password <*tws_user_password*>

   c. Stop and Restart of the Application Server
   d. changeTraceProperties.sh(.bat) -reset
   e. DeleteTrace.sh(.bat)

6. Which directory is *not* installed by Tivoli Workload Scheduler?
   a. /etc/Tivoli
   b. /etc/IBM
   c. /.swdis
   d. /usr/Tivoli

7. A client has installed a new instance of a Tivoli Workload Scheduler MDM that uses a remote Oracle 10g database. One day after the database server is rebooted, Tivoli Workload Scheduler operations requiring the database start failing, even though the DBA reports Oracle and the Oracle listener service are running fine. What should the client do?
   a. Restart the embedded WebSphere Express server
   b. Restart the Oracle database listener service
   c. Change the Oracle JDBC™ Connector Type in Tivoli Workload Scheduler to Type 4
   d. Change the Oracle Listener Port in TWS to match the new port reported by Oracle

8. What are the two methods of upgrading to a Version 8.4 MDM? (Select two.)
   a. Parallel
   b. Big bang
   c. Direct
   d. Cutover
   e. Delete old version

9. Which is *not* a supported UNinstallation method for a UNIX master domain manager?
   a. ISMP
   b. Silent ISMP
   c. twsinst
   d. Use rm -rf <*TWShome*>

10. A client uses Tivoli Workload Scheduler Version 8.3 that was installed with the Software Distribution CLI and wants to upgrade directly to Version 8.4. What should the Tivoli Workload Scheduler software package state be before starting the upgrade?

    a. UNDOABLE

    b. COMMIT

    c. ACCEPT

    d. REMOVE

    e. INSTALL

11. Once the FINAL job stream has been added to an “empty” Tivoli Workload Scheduler database, which script or command should be run to create the initial plan in Tivoli Workload Scheduler?

    a. composer CreatePlan

    b. conman CreatePlan

    c. JnextPlan

    d. JnextDay

12. Where does the **`makesec`** `input_security_text_file` command install the security file?

    a. In the *TWShome*/network/security file

    b. In the Tivoli Workload Scheduler global options table in RDBMS

    c. In the *TWShome*/mozart/Security file

    d. In the *TWShome*/Security file

13. Which command should be used to change the global configuration setting startOfDay?

    a. GlobalOpt chg

    b. **optman chg**

    c. Mozart chg

    d. optchg

14. Where is the netmth access method used to create inter-network dependencies in Tivoli Workload Scheduler?

    a. In *TWShome*/mozart

    b. In *TWShome*/methods

    c. In *TWShome*/stdlist

    d. In *TWShome*/network

15. A user wants to change the database name used in WebSphere Application Server on a UNIX master domain manager. Which set of WebSphere Application Server utilities can accomplish this?
    a. modifyTraceProperties.sh and changeTracePropserties.sh
    b. showHostProperties.sh and changeHostProperties.sh
    c. showDataSourceProperties.sh and changeDataSourceProperties.sh
    d. showSecurityProperties.sh and changeSecurityProperties.sh
16. Which databases are supported by the Tivoli Workload Scheduler engine? (Choose two.)
    a. DB2
    b. Microsoft SQL
    c. Oracle
    d. Sybase
17. Which one of the following is not a supported configuration?
    a. JSC V8.4 with MDM V8.3
    b. MDM V8.4 with Standard Agent V8.2.1
    c. TDWC V8.4 with MDM V8.2.1
    d. Agent V8.4 with MDM V8.2.1
18. Which of the following is not a supported operating system for the MDM?
    a. AIX
    b. HP-UX
    c. SUSE
    d. i5/OS
19. Which of the following are shipped with Tivoli Workload Scheduler? (Choose two.)
    a. DB2
    b. Oracle
    c. TEC
    d. WebSphere Application Server
20. What is the minimum amount of memory need to run the JSC?
    a. 256 MB
    b. 512 MB
    c. 1024 MB
    d. 2048 MB

Answers

The following list shows the correct answers to the sample questions in this appendix:

1. d
2. c
3. b, c
4. a
5. a, c
6. b
7. c
8. a, c
9. c
10. b
11. c
12. d
13. b
14. b
15. c
16. a, c
17. c
18. d
19. a, d
20. b

Related publications

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

IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 242. Some of the documents referenced here may be available in softcopy only.

- *Deployment Guide Series: IBM Tivoli Workload Scheduler V8.4 and IBM Tivoli Dynamic Workload Broker V1.2*, SG24-7528

Other publications

These publications are also relevant as further information sources:

- *IBM Tivoli Workload Scheduler Planning and Installation Guide V8.4* (revised March 2008), SC32-1273
- *IBM Tivoli Workload Scheduler for Applications User’s Guide V8.4,* SC32-1278
- *IBM Tivoli Workload Scheduler Job Scheduling Console User’s Guide V8.4*, SC32-1257
- *IBM Tivoli Workload Scheduler Reference Guide V8.4* (revised March 2008), SC32-1274
- *IBM Tivoli Workload Scheduler Administration and Troubleshooting V8.4 (revised March 2008),* SC32-1275
- *IBM Tivoli Dynamic Workload Console Installation and Troubleshooting Guide,* SC32-1572

Online resources

These Web sites are also relevant as further information sources:

- IBM Tivoli Workload Scheduler Infocenter:

  http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?toc=/com.ibm.tivoli.itws.doc/toc.xml

- Test 435 objectives:

  http://www-03.ibm.com/certify/tests/obj435.shtml

- Matrix showing the supported operating systems for all Tivoli products:

  http://www.ibm.com/software/sysmgmt/products/support/Tivoli_Supported_Platforms.html

How to get Redbooks

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

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

IBM

Redbooks

Certification Guide Series: IBM Tivoli Workload Scheduler V8.4

(0.5" spine)
0.475"<->0.875"
250 <-> 459 pages

Certification Guide Series: IBM Tivoli Workload Scheduler V8.4

Enables you to earn certification

Describes certification path and prerequisites

Provides sample test questions

This IBM Redbooks publication is a study guide for IBM Tivoli Workload Scheduler Version 8.4 and is aimed at readers who want to obtain IBM Professional Certification for this product.

The IBM Tivoli Workload Scheduler Version 8.4 Certification, offered through the Professional Certification Program from IBM, is designed to validate the required skills for technical professionals who implement the IBM Tivoli Workload Scheduler Version 8.4 product.

This book provides a combination of theory and practical experience required for a general understanding of the product. It also provides sample questions that help you evaluate your personal progress and provide familiarity with the types of questions that you encounter in the exam.

This publication does not replace practical experience, nor is it designed to be a standalone guide. Instead, when combined with education activities and experience, this book is an effective tool that can provide invaluable preparation for the exam.

For your convenience, we structure the chapters based on the sections of the IBM Tivoli Workload Scheduler V8.4 Implementation Certification test, such as planning, installation, and configuration. Each topic corresponds to one section of the exam and enables you to prepare for that section.

SG24-7628-00 ISBN 0738431397